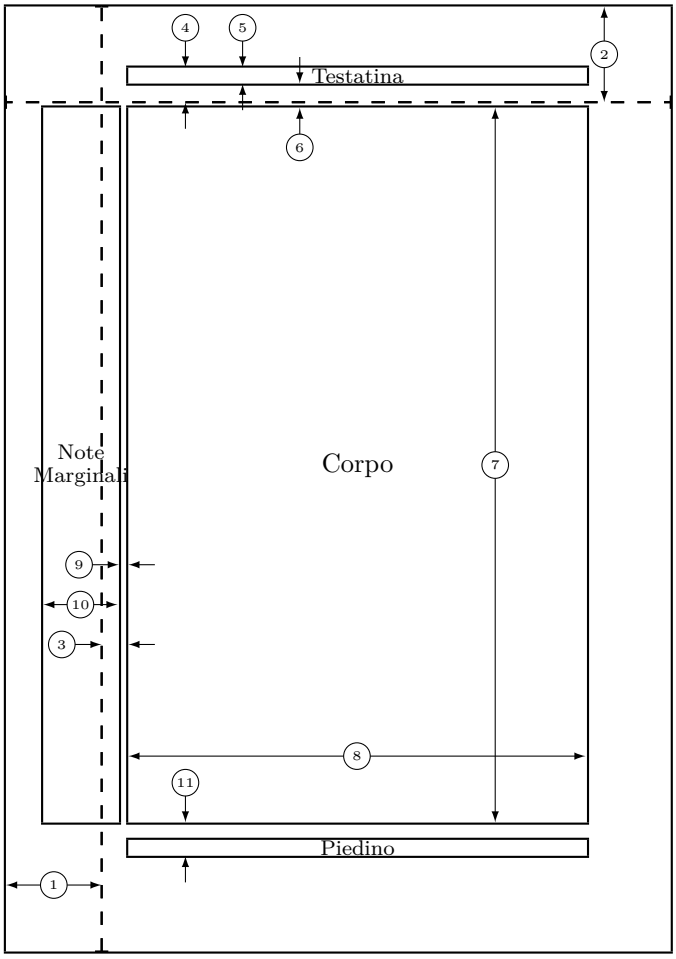
Gruppo Utilizzatori Italiani di TEX

Introduzione all'arte della composizione tipografica con LATEX

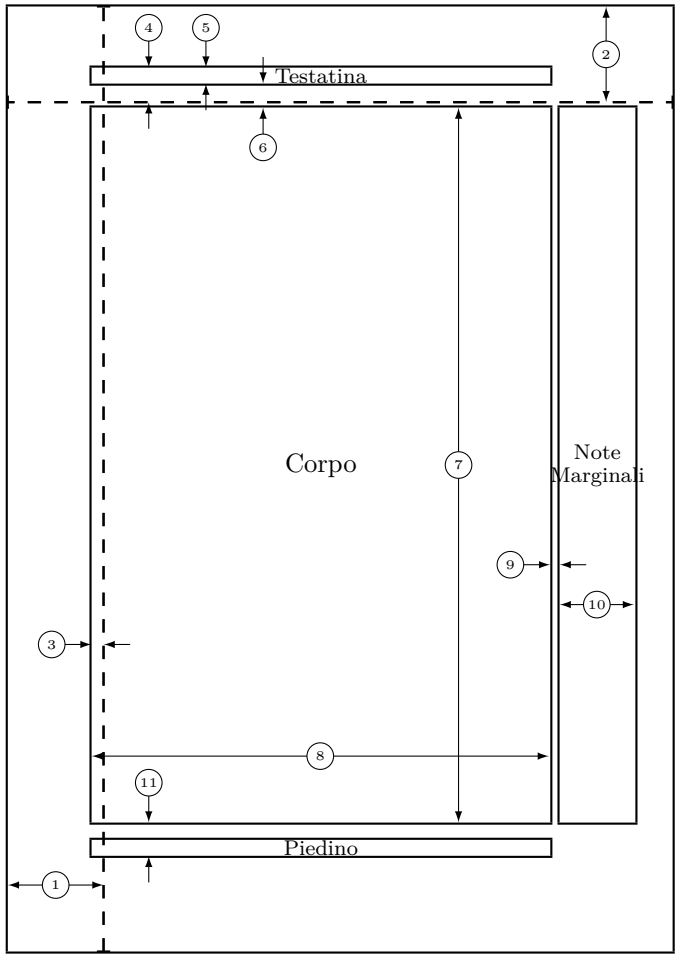
Tomo I

16 maggio 2015

Introduzione all'arte della composizione tipografica con LaTeX



1 un pollice + \hoffset
2 un pollice + \voffset
3 \evensidemargin = 20pt
4 \topmargin = -26pt
5 \headheight = 12pt
6 \headsep = 18pt
7 \textheight = 538pt
8 \textwidth = 345pt
9 \marginparsep = 7pt
10 \marginparwidth = 57pt
\marginparpush = 5pt (non mostrato)
11 \footskip = 25pt
\hoffset = 0pt
\voffset = 0pt
\paperwidth = 500pt
\paperheight = 711pt



1	un pollice + \hoffset	2	un pollice + \voffset
3	\oddsidemargin = -9pt	4	\topmargin = -26pt
5	\headheight = 12pt	6	\headsep = 18pt
7	\textheight = 538pt	8	\textwidth = 345pt
9	\marginparsep = 7pt	10	\marginparwidth = 57pt
11	\footskip = 25pt		\marginparpush = 5pt (non mostrato)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 500pt		\paperheight = 711pt

Introduzione all'arte della composizione tipografica con LaTeX

GuIt

Versione B5-0.99i — 2015/05/16
Pdf minor version = 5

Quest'opera è soggetta alla Creative Commons Public License versione 2.5 o posteriore. L'enunciato integrale della Licenza in versione 2.5 è reperibile all'indirizzo internet `http://creativecommons.org/licenses/by-nc-nd/3.0/deed.it`.

- Si è liberi di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera alle seguenti condizioni:

  Attribuzione Bisogna attribuire la paternità dell'opera nei modi indicati dall'autore o da colui al quale è stata data quest'opera in licenza; in questo caso si tratta del Gruppo degli Utilizzatori Italiani di TEX, GUIT.

  Non commerciale Non si può usare quest'opera per fini commerciali.

  Non opere derivate Non si può alterare o trasformare quest'opera, né usarla per crearne un'altra.

- Ogni volta che si usa o si distribuisce quest'opera, lo si deve fare secondo i termini di questa licenza, che va comunicata con chiarezza.

- In ogni caso si possono concordare con il titolare dei diritti d'autore (il GUIT, in questo caso) usi di quest'opera in deroga da questa licenza.

I nomi commerciali, i loghi, i trademark appartengono ai rispettivi proprietari.

La foto 2.2 della pagina 11 e le foto delle pagine 13, 229, e 432 — 434 sono di Claudio Beccari.
L'immagine 2.1 della pagina 11 è stata ricavata da un indirizzo Internet non più reperibile; l'autore sembra essere Hermann Zapf.

Coordinatore: Claudio Beccari

Hanno collaborato a questo testo in modo diretto o indiretto: Alessandro Andretta, Luciano Battaia, Francesco Biccari, Paolo Biffis, Riccardo Campana, David Carlisle, Antonio Cervone, Gustavo Cevolani, Agostino De Marco, Roberto Giacomelli, Tommaso Gordini, Enrico Gregorio, Orlando Iovino, Maurizio Himmelmann, Jerónimo Leal, Antonio Macrì, Giuseppe Molteni, Federico Morchio, Lorenzo Pantieri, Gianluca Pignalberi, Ottavio Rizzo, Salvatore Schirone, Luigi Scarso, Andrea Tonelli, Ivan Valbusa, Emanuele Zannarini.

Un grazie particolare a Enrico Gregorio.

Presentazione

Questo testo è rivolto a coloro che amano la bella composizione tipografica e trovano che i testi composti con LaTeX siano molto professionali. Questo implica che il lettore tipo abbia già un certa dimestichezza con LaTeX e abbia letto, almeno in parte, o il testo introduttivo [46] o, meglio ancora, l'ottimo manuale di Lorenzo Pantieri e Tommaso Gordini *L'arte di scrivere con LaTeX*, [49].

Alcune parti di questo testo possono risultare un po' ostiche a chi è completamente digiuno di programmazione, ma si è fatto ogni sforzo per rendere comprensibile queste parti anche a chi ha poca dimestichezza con il calcolatore. Certo c'è un compromesso fra chiarezza e concisione, ma si spera che il lettore vorrà ugualmente comprendere l'impegno del gruppo di persone che ha dato vita a questo testo.

Questo testo sulla composizione tipografica mediante LaTeX è stato infatti predisposto da diversi membri del Gruppo degli Utilizzatori Italiani di TeX, GuIt, il cui obiettivo è proprio quello di far conoscere il sistema di composizione sviluppato dal matematico Donald E. Knuth più di trenta anni fa. Non si tratta di un programma di interesse archeologico, perché esso è vivo e vegeto, ha dato origine a un buon numero di discendenti e alcune sue parti sono usate all'interno di altri programmi di elaborazione di testi, senza che gli utenti di questi programmi lo sappiano.

Il sistema TeX è stato uno dei primi esempi di software libero; questa sua qualità ne ha permesso il contributo creativo e/o critico di una moltitudine di utenti, come succede sempre con il software libero, per cui si è arricchito nel tempo di una moltitudine di estensioni che gli permettono di comporre praticamente qualsiasi cosa, tranne, forse, certi tipi di pieghevoli pubblicitari.

Si tratta di un programma di composizione tipografica, non di un impaginatore; quest'ultimo tipo di programmi consente di agire sul materiale da impaginare come se fosse una figura da modificare o da adattare, anche se svolge in parte le funzioni di compositore.

LaTeX, un elemento del sistema TeX, consente di comporre tipograficamente dei testi contenenti testo corrente, sia in prosa sia in poesia, scritti in qualunque alfabeto, per esempio latino, greco, cirillico, dall'andamento diretto (da sinistra a destra), oppure ebraico, arabo, dall'andamento inverso (da destra a sinistra), oppure cinese, giapponese, coreano, dall'andamento anche verticale.

Può gestire font di ogni genere, sia quelli a matrici di punti, sia i font PostScript, sia i font TrueType, OpenType, eccetera. Tali font possono essere codificati in varie maniere, ma alcuni 'figli' di TEX gestiscono anche i font codificati secondo la norma UNICODE.

La caratteristica che più differenzia il sistema TEX dagli altri elaboratori di composizione è il fatto che per comporre un documento con questo sistema bisogna agire in tempi diversi per introdurre il testo e per comporlo; in questo non è molto diverso da certi procedimenti professionali di impaginazione, dove il testo da comporre viene introdotto in un file di solo testo che poi viene in un secondo tempo fatto fluire dentro il programma di impaginazione, assemblandolo insieme alle figure e all'altro materiale non testuale per generare il documento 'finito', pronto da inviare alla fotoincisione e alla stampa.

Invece questo modo di comporre è molto diverso da quello dei word processor, dove il compositore vede direttamente sullo schermo del suo elaboratore il testo già composto, così da poter esaminare immediatamente il frutto del suo lavoro; l'analisi di questa differenza nel modo di procedere verrà svolta nel primo capitolo.

Il secondo capitolo darà al lettore alcune nozioni di tipografia, se non altro per abituarlo alle alcune parole che ricorrono spesso nella descrizione delle varie operazioni compositive. Chi avesse già queste nozioni può saltare la lettura di questo capitolo, ma se, nonostante tutto, decidesse di leggerlo, potrebbe constatare che alcune nozioni e/o alcuni vocaboli in questa guida sono usati per indicare cose leggermente diverse da quelle che conosceva.

Il terzo capitolo è dedicato all'ortografia tipografica; oltre alla definizione, che ovviamente non contrasta con l'ortografia della lingua, vengono impostate anche le buone regole per l'uso della sillabazione, delle forme dei caratteri, dell'uso della punteggiatura, dell'elisione e del troncamento, sottolineando che la grammatica consente numerose varianti, ma la buona tipografia predilige l'omogeneità e l'adesione a consuetudini ormai acquisite; la buona tipografia aiuta il lettore a leggere senza affaticarsi; usare convenzioni diverse dalle 'solite' affatica il lettore che deve abituarsi a comprendere quel che legge, ma che è scritto in modo per lui insolito.

Il quarto capitolo esaminerà le procedure da seguire per procurarsi il software del sistema TEX e per installare i programmi e i file accessori; non si vuole mettere in ombra la centralità dei programmi di composizione del sistema TEX, ma bisogna attrarre l'attenzione sui programmi accessori, per esempio gli *shell editor* la cui scelta può risultare determinante nella facilità con cui si può ottenente il meglio dal sistema TEX.

I capitoli cinque, sette ed otto richiamano i primi rudimenti della composizione asincrona mediante LATEX (parte del sistema TEX). Si parlerà essenzialmente di composizione testuale, di liste e di tabelle. Queste ultime già possono dare qualche pensiero anche al lettore con una modesta esperienza alle spalle.

LATEX si distingue da alcuni suoi fratelli, figli del sistema TEX, per consentire al compositore, oltre che allo scrittore, di concentrarsi sul messaggio da trasmettere al lettore, invece che sulla forma da dargli, cioè sulla sua estetica. A seconda del documento da comporre lo stile di composizione può essere molto diverso; anche

la semplice impaginazione può assumere aspetti grafici diversi; LaTeX consente di scegliere in modo globale il tipo di documento e di apportare piccole modifiche stilistiche durante la composizione, senza che il compositore debba preoccuparsi della nerezza dei caratteri, oppure della distanza del numero indicativo di un paragrafo dall'inizio del suo titolino. A tutte queste cose pensa la scelta iniziale del tipo di documento e con le opzioni specificate. Il capitolo sei si occupa appunto di queste cose.

I capitoli nove e dieci si dedicano alla creazione, manipolazione e inclusione di figure.

Passando ad argomenti più specializzati, i capitoli undici e dodici si occupano della preparazione della bibliografia e della composizione di indici analitici e glossari.

I capitoli tredici e quattordici si dedicano invece alla composizione della matematica; se il lettore ha familiarità con manuali di tipografia, avrà notato che l'argomento della composizione della matematica è praticamente assente da quei manuali, perché si tratta di un tipo di composizione abbastanza specializzato e che interviene raramente nei libri pubblicati. Ma un documento non è necessariamente un libro di narrativa; può essere un rapporto tecnico, una tesi di laurea o di dottorato, un manuale o un prontuario tecnico, un articolo scientifico, insomma, uno scritto dove la matematica compare spesso in forma avanzata.

Alla composizione di testi letterari e di filologia è dedicato il capitolo quindici; questi testi verosimilmente non contengono una sola formula matematica, ma richiedono un tipo di composizione spesso bidimensionale, che, come la matematica, richiede di comprendere a fondo i meccanismi compositivi bidimensionali. Alle necessità compositive della filologia si provvede mediante l'uso di moduli esterni, chiamati *pacchetti di estensione*, che fanno parte integrante del sistema TeX.

Il capitolo sedici si riferisce alla preparazione delle presentazioni, cioè di quei documenti che verranno verosimilmente usati per essere proiettati mentre se ne espone il contenuto a voce durante una conferenza, una lezione, o simili.

Il capitolo diciassette tratta dell'arte della composizione tipografica con LaTeX; questo è proprio il capitolo che giustifica il titolo di questo manuale; vi si parlerà delle tecniche di cui LaTeX o, più ancora, pdfLaTeX dispongono per eseguire la composizione tipografica come e meglio della totalità dei word processor e meglio di molti programmi di impaginazione; si tratta della microgiustificazione, una tecnica a cui ricorrevano fin dall'inizio i prototipografi, a cominciare da Gutenberg, ma di cui si è un po' persa l'abitudine nel corso dei secoli successivi, sia per i costi, sia per i vincoli imposti dai caratteri metallici; per fortuna la composizione elettronica è svincolata da queste limitazioni, ma ovviamente bisogna che il programma di composizione sia stato programmato per la microgiustificazione.

Il capitolo diciotto si riferisce ai caratteri da stampa e ne descrive le particolarità, i comandi per gestirli, i modi per caricare altri caratteri; fornisce qualche nozione di tipografia, ma dedica ampio spazio ai font del sistema TeX, il quale, è bene ribadirlo, non è limitato ai font disponibili in prima installazione, che

sono già molto numerosi, ma può usare virtualmente qualsiasi font disponibile sia gratuito, sia commerciale.

Il capitolo diciannove è dedicato all'arte di definire nuovi comandi compositivi per agevolare il compito del compositore.

Il capitolo venti tratta invece del progetto grafico delle pagine e delle strutture che compongono il documento; normalmente queste sono decisioni e scelte operate dal grafico editoriale; con LaTeX spesso siamo noi stessi i grafici editoriali, i compositori, talvolta anche gli editori dei nostri documenti; è bene avere anche una buona formazione in questo senso, almeno fino al livello di conoscere i problemi, senza avere la presunzione di saperli risolvere come un professionista. Conoscendo i problemi, però, può nascere il desiderio di conoscere più a fondo l'arte tipografica, che ci porterà non solo ad apprezzare maggiormente i prodotti tipografici ben riusciti, ma anche ad arricchire la nostra cultura in un settore spesso trascurato o considerato erroneamente troppo tecnico per meritare il nome di 'arte'. Ricordiamo invece che il nome del sistema TeX è ottenuto dalle prime tre lettere maiuscole della parola greca $\tau\acute{\epsilon}\chi\nu\eta$[1], che, come ricorda Knuth stesso, vuol dire *arte*.

Questo testo non è un manuale; per LaTeX ce ne sono di liberi e di commerciali assai validi; a questo argomento è dedicata il capitolo ventuno. Nello stesso tempo, dicendo che questo non è un manuale si vorrebbe sottolineare che le indicazioni qui esposte servono per affrontare la composizione con LaTeX con un approccio che guarda di più alla sostanza, alla composizione professionale, che non all'aspetto grafico del singolo dettaglio man mano che questo si presenta, come succede spesso, invece, con i vari word processor.

Per esempio in questo testo si parla di 'scatole' proprio di sfuggita, mentre in ogni manuale questo argomento richiede almeno una sezione espressamente dedicata loro. Così si parla poco o nulla di contatori o di lunghezze rigide o elastiche; sono argomenti importanti, ma che servono maggiormente per scrivere i programmi, le macroistruzioni, contenute nei file che specificano lo stile compositivo o nei file che raccolgono le macro personali. Durante la composizione non bisognerebbe mai perdersi in questi dettagli.

Piuttosto la programmazione in linguaggio TeX può diventare essenziale per l'utente che deve comporre testi dallo stile insolito o che abbia bisogno di strutture compositive particolari.

Qui, però, tranne una esposizione sommaria nel capitolo 19, non si parlerà di queste cose, ma si rinvia direttamente il lettore a manuali che trattano questo aspetto con maggiore dettaglio, (capitolo 21). Si veda piuttosto il capitolo 29, di tipo più manualistico, dove sono raccolte, commentate, modificate le informazioni che Leslie Lamport ha scritto nel suo manuale [35].

Il capitolo ventidue è dedicato ad un aspetto piuttosto moderno e fortemente legato alla tipografia elettronica, vale a dire si occupa del problema dell'archiviabilità dei documenti elettronici; questa caratteristica richiede strumenti ed

[1] $\tau\acute{\epsilon}\chi\nu\eta$ in Italia viene pronunciata 'tècne'; in Grecia 'téchni'; in ogni caso quella che sembra una X nel nome di TeX e di LaTeX va pronunciata come una 'k' o come una forte espirazione, non come una 'x'.

accorgimenti particolari che si possono raggiungere con l'uso del sistema TEX, ma che richiedono comunque una speciale attenzione.

Il capitolo ventitre riguarda invece un aspetto produttivo che coinvolge la tipografia elettronica; si tratta del lavoro di assemblaggio dei file da comporre con il sistema TEX, segnatamente con LATEX e pdfLATEX, quando i file sorgente non hanno il mark-up di LATEX. Si può eseguire l'elaborazione manuale, ma è meglio disporre di adeguati strumenti per la conversione automatica, almeno per ottenere un primo approccio al mark-up di LATEX, ma con la trasformazione delle codifiche dei vari caratteri in modo da non alterare il messaggio del testo da comporre e, specialmente, per non introdurre errori durante il processo di conversione. L'editoria moderna deve poter usare questi strumenti con cognizione di causa, sia per il lavoro commerciale sia per quello che gradatamente si va estendendo, cioè la documentazione libera, la produzione di testi liberamente scaricabili dalla rete, ma composti con ogni cura al massimo livello qualitativo.

Il capitolo 24 è dedicata alla simbologia e alla nomenclatura delle grandezze fisiche, nonché ai simboli codificati dalle norme ISO per l'uso nella matematica usata dai fisici e dai tecnologi; vi sono anche non pochi riferimenti al Sistema Internazionale delle Unità di Misura e alla loro 'ortografia'.

Il capitolo 25 spiega invece in ogni dettaglio l'algoritmo che viene usato dai programmi del sistema TEX per dividere le parole in fin di riga; questo è un procedimento che in italiano funziona impeccabilmente nella totalità dei casi, ma è bene conoscerne i dettagli per intervenire in quei pochi casi in cui la giustificazione non viene eseguita perfettamente oppure quando si scrive in una lingua straniera dalle regole più rigide di quelle che valgono per l'italiano; oppure per ricorrere ad una sillabazione italiana diversa da quella prescritta dalla norme UNI, ma consentita in casi particolari.

Il capitolo 26 vorrebbe riempire un vuoto di informazione che per usare LATEX va assolutamente colmato. Si tratta della distinzione fra la codifica di entrata e di quella di uscita; fra la codifica che l'utente, tramite i suoi programmi, usa per scrivere un testo usando la tastiera, e quella che il programma di composizione usa in relazione ai font impiegati. Bisogna conoscere queste due codifiche, distinguerle e usarle con cognizione di causa.

Il capitolo 27, molto tecnico, cerca di spiegare i dettagli di come ogni motore di composizione del sistema TEX costruisca i capoversi e costruisca le pagine; si tratta di un meccanismo abbastanza complesso, ma la sua comprensione aiuta a capire perché la composizione talvolta sia diversa da come ce la aspetteremmo. Normalmente ha ragione il motore di composizione a fare quello che fa, ma talvolta può essere necessaria una piccola dose di aiuto consapevole.

Il capitolo 29, lungi dal rappresentare una semplice traduzione della corrispondente appendice del manuale [35] di Leslie Lamport, il creatore di LATEX, pur mantenendone l'impostazione, rappresenta, si spera, una fonte comoda e pratica per rivedere in modo ragionato ma non troppo prolisso la sintassi dei vari comandi del mark-up di LATEX. In questa parte, per la verità, si sono saltate rispetto al manuale di Lamport, intere parti che oggi sono considerate superate; invece alcune parti sono state estese considerevolmente.

Il lettore non si scoraggi se vengono usate molte parole inglesi; in un certo senso la cosa è voluta, perché i comandi del *mark-up* di LaTeX sono quasi tutti formati da parole inglesi o da loro abbreviazioni; in ogni punto in cui questi comandi vengono descritti si è sempre cercato di associare le parole inglesi alla spiegazione italiana; tuttavia non si è rifuggiti da alcune similitudini insolite, come 'flottante' (che esiste in italiano con un significato simile) corrispondente a *floating*, ritenuto preferibile rispetto all'italianissima parola 'mobile' che esprime lo stesso concetto, ma che forse è espresso con una parola slegata dai messaggi che LaTeX espone sullo schermo durante la sua esecuzione. Così si è frequentemente usata la locuzione *di default*, piuttosto che usare espressioni italiane come 'predefinito', 'da usarsi in mancanza d'altro', 'se non si è specificato diversamente', e simili locuzioni quasi equivalenti l'una all'altra.

Inoltre, sebbene non si sia sicuri di essere stati sempre consistenti, è stata usata la congiunzione oppositiva 'o' per indicare il concetto esclusivo di 'o questo o quello ma non entrambi', e la congiunzione 'oppure' per indicare il concetto inclusivo di 'o questo o quello o entrambi'. La lingua parlata e quella scritte sono flessibili e talvolta non è necessario essere così precisi, ma si cerchi di tenere presente questo particolare uso delle congiunzioni oppositive.

Di fianco a molte figure vi è una piccola legenda, spesso in verticale, con il nome dell'autore e il nome del programma, quasi sempre del sistema TeX, che l'autore ha usato. Questa indicazione non è messa vicino alle fotografie oppure alle immagini tratte da Internet; la paternità di queste immagini è indicata, ove possibile, nel retrofrontespizio. Per le altre figure, oltre a indicare l'autore del disegno mostrato, si è ritenuto utile indicare anche il programma usato per realizzare il disegno; quasi tutti i disegni sono stati realizzati usando comandi e mark-up forniti da pacchetti di estensione del programma di composizione; questi pacchetti consentono di eseguire egregiamente molti tipi di disegni, senza bisogno di ricorrere a programmi esterni; hanno anche il vantaggio di essere stilisticamente compatibili con la tipografia eseguita con LaTeX, cosa che spesso non è possibile ottenere con i normali word processor o con altri programmi di disegno assistito dal calcolatore.

Il materiale di questa guida è stato riordinato rispetto alla edizioni precedenti; ovviamente è anche stato aggiornato e l'aggiornamento continuo cerca di non restare indietro rispetto all'evoluzione continua del sistema TeX; non è possibile garantire che questo testo sia sempre aggiornato all'ultima distribuzione del sistema TeX, ma si fa il possibile per farlo.

Questa guida può essere composta in formato A4 (210 mm per 297 mm) o in formato B5 (176 mm per 250 mm); quest'ultimo formato può venire stampato e rifilato in modo che il volume brossurato a colla o legato possa trovare facilmente spazio nei normali scaffali; può venire anche stampato in tre tomi distinti in formato B5, dove però le pagine iniziali (dal frontespizio fino agli indici) e le pagine finali (dalla bibliografia in poi) siano comuni a tutti i tomi; i riferimenti degli indici generale e analitici saranno quindi sempre corretti anche in riferimento a tomi diversi.

Indice

Presentazione IX

1 Composizione sincrona e asincrona 1
1.1 Introduzione . . . 1
1.2 Il mark-up di LaTeX . . . 2
1.3 Scribus: un impaginatore libero con licenza GPL . . . 3
1.4 Conclusione . . . 7

2 Nozioni elementari di tipografia 9
2.1 Tipografia e dattilografia . . . 9
2.2 Unità di misura tipografiche . . . 10
2.3 Misure tipografiche . . . 13
2.4 Le particolarità dei caratteri . . . 15
2.5 I contrografismi . . . 17
2.6 Le parti di alcuni documenti a stampa . . . 18
2.7 Osservazioni finali . . . 20

3 Ortografia tipografica 21
3.1 Ortografia testuale . . . 21
3.1.1 Ortografie alternative . . . 22
3.1.2 'd' eufonica e 'i' prostetica, apostrofi . . . 23
3.2 Accenti . . . 24
3.3 Sillabazione . . . 26
3.4 Punteggiatura . . . 27
3.5 Abbreviazioni . . . 33
3.6 Appellativi e maiuscole . . . 35
3.7 Uso dei font . . . 36
3.8 Le note . . . 37
3.9 Conclusioni . . . 38

4 Installare il sistema TeX 41
4.1 Installazione su macchine Windows da XP in poi . . . 42
4.1.1 Installare MiKTeX . . . 42

4.1.2 Installare TEX Live 43
4.2 Installazione su Linux 44
4.3 Installazione su Macintosh con Mac OS X 45
4.4 Gli alberi di cartelle del sistema TEX 48
4.4.1 Gli alberi di cartelle di TEX Live 50
4.4.2 Gli alberi di cartelle di MiKTEX 51
4.4.3 L'aggiornamento dei database dei nomi dei file 53
4.5 I programmi accessori 54
4.6 Le tastiere 56
4.6.1 Le tastiere delle macchine Windows 57
4.6.2 Tastiere Linux 60
4.6.3 Le tastiere sulle macchine Macintosh 60
4.6.4 Le tastiere virtuali 61
4.7 Gli shell editor 62
4.7.1 Shell editor multipiattaforma 65
4.7.2 Shell editor per le macchine Windows 68
4.7.3 Shell editor per le macchine Linux 70
4.7.4 Shell editor per le macchine Macintosh 73
4.8 Editor quasi WYSIWYG 77
4.9 LATEX e pdfLATEX 78
4.10 Altri programmi del sistema TEX 81
4.10.1 Plain TEX 82
4.10.2 I programmi estesi 82
4.10.3 Il mark-up ConTEXt 84
4.10.4 I programmi Omega e Lambda, Aleph e Lamed 85
4.10.5 Il programma XETEX 85
4.10.6 Il programma LuaTEX 86
4.11 Il sistema TEX 88

5 LATEX: prime nozioni 93
5.1 Introduzione 93
5.2 L'inizio del file sorgente 93
5.3 Il documento 95
5.4 La fine del documento 96
5.5 Un semplice esercizio 96
5.6 I caratteri speciali 98
5.7 Testo strutturato 100
5.8 Organizzazione dei file sorgente 101
5.9 Gestione degli errori 108

6 LATEX: i vari tipi di documenti e stili di composizione 111
6.1 Introduzione 111
6.2 Classi standard 111
6.3 La creazione di nuove classi 113
6.4 Alcune classi non standard 114

6.4.1 Le classi Komascript 114
6.4.2 La classe *memoir* 115
6.4.3 La classe *ncc* 116
6.4.4 Le classi per le tesi di laurea 117
6.4.5 L'estensione `layaureo` 119
6.5 I pacchetti di estensione 120
6.5.1 Come invocare i file di estensione 121
6.5.2 I vari pacchetti e gli archivi internazionali 122
6.6 Come scrivere nuovi pacchetti 123
6.7 Non modificare i pacchetti esistenti 126

7 LaTeX: testi speciali 131
7.1 Che cosa sono i testi in display 131
7.2 Le citazioni 131
7.2.1 Le citazioni brevi 132
7.2.2 Le citazioni lunghe 133
7.2.3 I versi 134
7.2.4 Brani in linguaggi speciali 135
7.3 Gli elenchi 136
7.3.1 Le elencazioni in linea 136
7.3.2 Le enumerazioni 136
7.3.3 Le elencazioni semplici 137
7.3.4 Alcune osservazioni relative alle elencazioni 138
7.4 Le descrizioni 139
7.5 Le liste bibliografiche 139
7.6 I riferimenti incrociati 141
7.7 Altri testi in display 142
7.8 Le note 145
7.8.1 Le note in calce 145
7.8.2 Le note marginali 147
7.9 Un esempio specifico di testi speciali 148

8 LaTeX: tabelle 153
8.1 Introduzione 153
8.2 Come far flottare una tabella 154
8.3 Le didascalie 154
8.4 Come comporre la tabella vera e propria 155
8.4.1 I descrittori delle colonne 156
8.4.2 Il raggruppamento delle celle 157
8.4.3 I separatori verticali 159
8.4.4 Come rendere le tabelle un poco più aperte 160
8.5 Le tabelle di larghezza specificata 162
8.6 Problemi compositivi delle tabelle 165
8.6.1 Tabelle troppo larghe 165
8.7 Tabelle troppo lunghe 167

8.8 Pacchetti di estensione per le tabelle 168

9 LaTeX: figure **175**
9.1 Le figure e le immagini 175
9.2 L'ambiente `figure` 175
9.2.1 Controllo dei grandi oggetti flottanti 176
9.2.2 Modifica degli ambienti flottanti 176
9.3 L'ambiente `picture` 177
9.4 Il pacchetto **pgf** 185
9.5 Vantaggi dei programmi nativi del sistema TeX 188
9.6 METAPOST 189
9.7 Usi insoliti dell'ambiente *picture* 191
9.8 Linee guida per la grafica 193
9.8.1 Preliminari 194
9.8.2 Il tempo necessario per la creazione della grafica 195
9.8.3 Piano di lavoro per creare un grafico 196
9.8.4 Collegamento fra testo e grafico 196
9.8.5 Coerenza fra testo e grafica 198
9.8.6 Legende nei grafici 199
9.8.7 Diagrammi di vario genere 199
9.8.8 Attenzione e distrazione 205
9.8.9 Commenti 206

10 LaTeX: l'importazione di figure esterne **209**
10.1 Introduzione 209
10.2 I formati grafici 209
10.2.1 I formati vettoriali 209
10.2.2 I formati diversi da quelli vettoriali 210
10.3 I formati accettabili 212
10.3.1 I formati accettabili da latex 214
10.3.2 I formati accettabili da pdflatex 216
10.4 Conversione dei formati 218
10.5 Scontornare le immagini 220
10.6 L'importazione delle immagini 221
10.6.1 Organizzare le immagini 222
10.6.2 Includere le immagini 225

11 La bibliografia **231**
11.1 Introduzione 231
11.2 I programmi di estrazione bibliografica 232
11.2.1 Come specificare lo stile bibliografico 232
11.2.2 Come comporre la bibliografia 232
11.2.3 Chiavi e citazioni 233
11.3 I database bibliografici 235
11.4 Quanti database bibliografici? 237

11.5 Il pacchetto *biblatex* 239

12 LaTeX: indici e glossari **241**
12.1 Introduzione 241
12.2 L'indice analitico 241
12.2.1 Il programma makeindex 242
12.2.2 La composizione effettiva dell'indice analitico 243
12.3 Il glossario 244
12.4 Modifica dell'indice analitico 245
12.5 Indicizzazione sincrona 245
12.5.1 Indicizzazione sincrona dell'indice analitico 247
12.5.2 Indicizzazione sincrona del glossario 248
12.6 Composizione automatica dell'indice analitico 248

13 LaTeX: la matematica semplice **251**
13.1 Introduzione 251
13.2 I modi matematici 251
13.3 Alcune annotazioni sulle lettere greche 256
13.4 Alcune osservazioni sugli operatori funzionali 257
13.5 Alcune osservazioni sui grandi operatori 260
13.6 I grandi delimitatori 261
13.7 Accenti e segni diacritici matematici 262
13.8 Gli ambienti matematici 263
13.9 Le unità di misura 265

14 LaTeX: la matematica avanzata **269**
14.1 I simboli di *amsmath* 269
14.2 Le estensioni dei font matematici 272
14.3 I sistemi di equazioni 272
14.4 Gli ambienti di composizione di *amsmath* 273
14.4.1 L'ambiente *equation* 274
14.4.2 L'ambiente *aligned* 274
14.4.3 L'ambiente *split* 275
14.4.4 L'ambiente *multline* 276
14.4.5 L'ambiente *gather* 277
14.4.6 L'ambiente *align* 277
14.4.7 L'ambiente *flalign* 278
14.4.8 L'ambiente *alignat* 279
14.4.9 L'ambiente *subequations* 279
14.4.10 Gli ambienti spezzati 280
14.5 Altri comandi e ambienti 281
14.5.1 Definizione di operatori funzionali 281
14.5.2 Le frazioni in generale e le frazioni continue 282
14.5.2.1 Le frazioni e gli altri costrutti simili 282
14.5.2.2 Le frazioni continue 282

14.5.3 Il testo intercalato alle equazioni 282
14.5.4 Le frecce estensibili 284
14.5.5 Gli indici incolonnati 284
14.5.6 Gli integrali multipli 284
14.5.7 L'operatore differenziale 285
14.5.8 I simboli corsivi matematici in nero 285
14.5.9 Le espressioni matematiche riquadrate 286
14.6 Le matrici e i determinanti 287
14.7 I diagrammi commutativi 290
14.8 La punteggiatura in matematica 291
14.9 Conclusioni 295

15 LaTeX: composizione di testi letterari e filologici 297
15.1 La composizione di testi letterari 297
15.2 Scrivere in greco 301
15.3 Font adatti ai testi classici 304
15.4 La composizione del tedesco classico 305
15.5 La composizione di testi in lingue classiche 307
15.5.1 Scrivere in latino 307
15.5.2 Scrivere in greco 308
15.6 La composizione di testi filologici 311
15.7 Un esempio di composizione con XeLaTeX 314
15.7.1 Esempio di composizione in greco 314
15.7.2 XeLaTeX, i font OpenType e i font Type 1 316
15.8 Conclusione 319

16 LaTeX: le presentazioni 321
16.1 Introduzione 321
16.2 Le classi per le presentazioni 321
16.3 Altre classi per le presentazioni 323
16.4 La classe `beamer` 323
16.5 La documentazione 324
16.6 Una breve presentazione 325
16.7 Creare un nuovo stile 331
16.8 Osservazioni 332

17 LaTeX: la microgiustificazione 333
17.1 Introduzione 333
17.2 La composizione dei capoversi 334
17.3 Metodi per migliorare la giustificazione 336
17.4 La microgiustificazione 337
17.5 Come funziona la microtipografia 339

18 LaTeX: i caratteri da stampa **343**
18.1 Introduzione . . . 343
18.2 Terminologia relativa ai caratteri . . . 343
18.3 I comandi per la scelta dei font . . . 352
18.3.1 La scelta del corpo e dell'avanzamento di riga . . . 352
18.3.2 La scelta delle altre caratteristiche . . . 357
18.4 Altri font diversi da quelli di default . . . 359
18.5 I font per la matematica . . . 360
18.6 Il Text Companion Font . . . 366
18.7 Gli alfabeti diversi da quello latino . . . 367
18.7.1 Caratteri cirillici . . . 367
18.7.2 Caratteri greci . . . 369
18.7.3 Scrivere con altri alfabeti . . . 371
18.8 La gestione dei font . . . 371
18.8.1 Altri font già disponibili . . . 375
18.9 Installazione di altri font vettoriali . . . 377
18.9.1 Operazioni preliminari . . . 379
18.9.2 Installare un font Type 1 creato con FontForge . . . 381
18.9.3 Installare un font Type 1 creato con METAFONT . . . 382
18.9.4 Installare font Type 1 . . . 383
18.9.5 Installare font TrueType . . . 384
18.9.6 Installare font OpenType . . . 384
18.9.7 Aggiornamento dei file generali di mappa . . . 384
18.9.7.1 TeX Live . . . 384
18.9.7.2 MiKTeX . . . 386
18.9.8 I file di descrizione dei font . . . 387
18.9.9 Ricapitolazione . . . 392
18.9.10 Font e sistema TeX . . . 393
18.10 Conclusioni . . . 397

19 LaTeX: nuovi comandi **399**
19.1 Introduzione . . . 399
19.2 Le definizioni di comandi nuovi . . . 400
19.3 Ridefinizione di comandi già esistenti . . . 404
19.4 Ridefinizioni di comandi di sistema . . . 405
19.4.1 I numeri romani maiuscoletti . . . 405
19.4.2 La virgola intelligente . . . 410
19.4.3 Il numero nelle liste delle tabelle e delle figure . . . 415
19.5 Esiste già o non esiste ancora il comando? . . . 416
19.6 Definizione di comandi robusti . . . 417
19.7 Definizione di un nuovo ambiente . . . 418
19.8 La ridefinizione di ambienti esistenti . . . 420
19.9 Situazioni particolari . . . 424
19.9.1 Le linee guida . . . 424
19.9.2 Controllo della posizione di grandi oggetti . . . 427

19.9.3 Immagini, celle e scatole 428

20 LaTeX: la geometria delle pagine 437
20.1 Introduzione 437
20.2 La geometria della pagina 439
20.2.1 Il formato delle pagine 439
20.2.2 Le segnature e le imposizioni 441
20.2.3 I crocini 443
20.2.4 Dimensioni della gabbia del testo 444
20.2.4.1 Testatine e piedini; il pacchetto *fancyhdr* . . . 446
20.2.4.2 Le proporzioni della gabbia di testo 448
20.2.4.3 I margini 449
20.3 Lo scartamento e i contrografismi verticali 454
20.4 I capoversi 457
20.5 Testatine e piedini 459
20.6 I pacchetti di personalizzazione 460
20.6.1 Il pacchetto *geometry* 460
20.6.2 I pacchetti *titlesec*, *fancyhdr*, *sectsty* e *tocloft* 464
20.6.3 Testatine 466
20.7 La pagina del titolo 469
20.8 Le inserzioni flottanti e non flottanti 470
20.8.1 Gli oggetti flottanti 470
20.8.2 Gli oggetti non flottanti 471
20.9 Conclusioni 473

21 Dove documentarsi 475
21.1 La documentazione essenziale 475
21.2 Documentazione sulla tipografia 476
21.3 Documentazione su LaTeX 479
21.4 Documentazione sulla grafica 480
21.5 Documentazione sui singoli pacchetti 480
21.6 Documentazione su TeX 481
21.7 Documentazione sui simboli di LaTeX 482
21.8 Composizione della matematica 482
21.9 L'archiviazione dei documenti 484

22 Il formato PDF archiviabile 487
22.1 Le immagini 488
22.2 I font 490
22.3 Gli hyperlink 493
22.4 Generazione di un file PDF archiviabile 493
22.4.1 La strada maestra 493
22.4.2 Trasformazione di un file PDF o di un file PS 495
22.5 Conclusioni 498

23 Comporre documenti di molti autori **499**
23.1 Conversione manuale 500
23.1.1 Copia e incolla 501
23.2 Conversione automatica 507
23.2.1 Documenti in formato `.doc` 507
23.2.2 Documenti in formato PDF 510
23.2.3 Documenti in formato XML 512
23.3 Documenti in collaborazione 513
23.3.1 GIT 515
23.3.2 Organizzazione minimale 515
23.3.2.1 File `txt` 515
23.3.2.2 File MMD e MD 516
23.3.3 Andare oltre: Scrivener 517
23.3.3.1 Compilazione del documento finale in PDF . . 520
23.3.3.2 Compilazione del documento finale in file `tex` . 522
23.3.4 Commenti in stile *html* 522
23.4 Conclusioni 524

24 Simbologia matematica e fisica **527**
24.1 Unità di misura del Sistema Internazionale 527
24.2 Simboli matematici nelle scienze 530
24.3 Nomenclatura 532

25 Divisione in sillabe **557**
25.1 Quando viene eseguita la cesura 557
25.2 La sillabazione fonetica oppure etimologica 561
25.3 Come fa TEX a dividere in sillabe 562
25.4 Bruttezza residua 565
25.5 I pattern per la lingua italiana 566
25.6 Come rimediare a righe sporgenti dalla giustezza 568
25.7 I file di pattern 569

26 Codifica in entrata e codifica in uscita **575**
26.1 Introduzione 575
26.2 Le tre distinte codifiche di TEX 578
26.2.1 La codifica di ingresso 586
26.2.2 La codifica di uscita 588
26.3 Specificare la codifica giusta 590
26.3.1 Scoprire la codifica di input 592
26.3.2 Cambiamento della codifica 594
26.4 Collage di contributi diversi 598
26.5 Considerazioni riassuntive 598

27 Come fa TEX a comporre le pagine **601**
27.1 Divisione dei capoversi 602
27.2 Divisione della pagine 604
27.3 Cosa fare se... 606
27.3.1 Collocazione degli oggetti flottanti 606
27.3.2 Le equazioni ingombranti 607
27.3.3 I sezionamenti 608
27.4 Conclusioni 609

28 Trattamento degli errori **611**
28.1 Errori ortografici nei nomi dei comandi 611
28.2 Errore nella ricerca dei file 611
28.3 Ciclo infinito 613
28.4 Gruppi aperti 614
28.5 Mancata apertura di un gruppo 615
28.6 Interferenza con le definizioni 616
28.7 File personali che provocano conflitti 616
28.8 Tracciare l'operato del programma 618
28.8.1 I comandi primitivi di tracciamento 618
28.8.2 Il pacchetto *trace* 619
28.9 Costruzione dei capoversi e delle pagine 622
28.10 Conclusioni 625

29 Riepilogo della sintassi di LATEX **627**
29.1 La struttura del documento 629
29.2 Periodi e capoversi 630
29.2.1 Periodi 630
29.2.2 Capoversi 631
29.2.3 Note in calce 633
29.2.4 Note marginali 633
29.2.5 Accenti e simboli speciali 634
29.3 Suddivisione del testo e indici 634
29.3.1 Comandi di sezionamento 634
29.4 Classi, pacchetti e stili delle pagine 638
29.4.1 Classe del documento 638
29.4.2 Pacchetti 640
29.4.3 Stili delle pagine 641
29.4.4 Il frontespizio 643
29.5 Testi in display 645
29.5.1 Citazioni e poemi 645
29.5.2 Liste 645
29.5.3 Testo composto verbatim 650
29.6 Formule matematiche 651
29.6.1 Formule 651
29.6.2 Simboli, accenti, delimitatori e grandi operatori 654

29.6.3 Impilare gli oggetti matematici 654
29.6.4 Spaziatura matematica 654
29.6.5 Font matematici 655
29.6.6 Stili di composizione 656
29.7 Definizioni, numeri e programmazione 656
29.7.1 Comandi di definizione 656
29.7.2 Comandi per la definizione di ambienti 657
29.7.3 Teoremi 657
29.8 Numeri, lunghezze e spazi 658
29.8.1 Numeri 658
29.8.2 Lunghezze 662
29.8.3 Operazioni fra numeri e grandezze 663
29.8.4 Il pacchetti *ifthen* e *etoolbox* 667
29.8.4.1 Il pacchetto *ifthen* 667
29.8.4.2 Il pacchetto *etoolbox* 670
29.9 Spaziature 673
29.10 Figure, tabelle ed altri oggetti flottanti 674
29.10.1 Figure e tabelle 674
29.10.2 Note marginali 679
29.11 Incolonnamenti 681
29.11.1 L'ambiente tabbing 681
29.11.2 Gli ambienti array e tabular 682
29.12 I file ausiliari e i loro comandi 685
29.12.1 I file del sistema TEX 685
29.12.2 I riferimenti incrociati 686
29.12.3 Bibliografia e citazioni 686
29.12.4 Suddivisione del file sorgente 687
29.13 Indice analitico e glossario 688
29.13.1 Indice analitico 688
29.13.2 Glossario 689
29.14 Compilazione interattiva 689
29.15 Interruzione di riga e di pagina 690
29.15.1 Interruzione di riga 690
29.15.2 Interruzione di pagina 691
29.16 Scatole 692
29.16.1 Scatole di uso immediato 693
29.16.2 Scatole per conservare del testo 694
29.16.3 Ambienti e comandi per scatole particolari 695
29.16.4 I comandi per le scatole con il linguaggio di TEX 697
29.16.5 Operazioni di misura sulle scatole 700
29.17 Disegni e colori 701
29.17.1 Disegni 701
29.17.2 Colori e grafica 701
29.18 Selezione dei caratteri 704
29.18.1 Scegliere famiglia, forma e serie 704

29.18.2 Scegliere il corpo 705
29.18.3 Corpi testuali e matematici 705
29.18.4 Simboli speciali 706

Bibliografia **707**

Indice analitico **712**

Indice degli ambienti **728**

Indice delle classi **730**

Indice dei file **731**

Indice dei pacchetti **734**

Indice dei programmi e delle distribuzioni **737**

Elenco delle tabelle

4.1 Legame fra alcuni comandi di sistema, gli interpreti e i formati . 81

8.1 Descrittori delle colonne per le tabelle 156
8.2 Tabella la cui colonna di destra è specificata con `p{80mm}` . . . 158
8.3 Tabella adattata alla giustezza 163
8.4 Tabella male adattata alla giustezza 163
8.5 Tabella bene adattata alla giustezza 163
8.6 Tabella composta con le estensioni del pacchetto *array* 170

9.1 Tabella insolita 192
9.2 Esempio di valutazione di un seminario 200

10.1 Dimensioni di `texbook.pdf` con vari metodi 215

13.1 Le lettere greche 254
13.2 Gli operatori di relazione 255
13.3 Gli operatori binari 255
13.4 Gli operatori funzionali 256
13.5 I grandi operatori 256
13.6 I grandi delimitatori 257
13.7 Accenti e diacritici matematici 258
13.8 Altri simboli 258

14.1 Prima serie di simboli accessibili con il pacchetto *amsmath* . . . 270
14.2 Seconda serie di simboli accessibili con il pacchetto *amsmath* . 271
14.3 Gli ambienti di allineamento di *amsmath* 273

15.1 Formati britannici della pagina 298
15.2 I tasti della tastiera latina e i segni dell'alfabeto greco 302

18.1 Istruzioni per la scelta del corpo dei caratteri 353
18.2 I corpi ottici a confronto col corpo unico 354
18.3 Il font latino a 128 caratteri con codifica OT1 355
18.4 Il font cirillico a 128 caratteri con codifica OT2 355

18.5 Il font latino a 256 caratteri con codifica T1 356
18.6 Varie combinazioni di serie e forma per i caratteri standard . . . 358
18.7 Il corsivo matematico con codifica OML 362
18.8 La polizza dei simboli matematici con codifica OMS 362
18.9 I segni matematici estensibili con codifica OMX 363
18.10 Il Text Companion Font con codifica TS1 366
18.11 Il font cirillico esteso con la codifica *X2* 368
18.12 Il font greco a 256 caratteri con codifica LGR 370
18.13 Il font latino con codifica LY1 378

19.1 Allineamento scorretto in una tabella 432
19.2 Allineamento corretto in una tabella 432
19.3 Tabella con allineamento corretto e spaziatura corretta 432
19.4 Allineamento aggiustato in modo accurato 434

20.1 Formati delle pagine a seconda della segnatura 440
20.2 Giustezze determinate mediante la lunghezza dell'alfabeto . . . 447

24.1 Unità fondamentali 527
24.2 Prefissi decimali 528
24.3 Prefissi binari 528
24.4 Unità logaritmiche 528
24.5 Unità derivate 530
24.6 Unità di misura legalmente ammesse 531
24.7 Unità di misura temporaneamente accettate 532
24.8 Simboli matematici 533
24.9 Nomenclatura, simboli e unità di misura 547

25.1 Pattern usati per dividere in sillabe *dell'istruzione* 563
25.2 Pattern usati per dividere in sillabe *discinesia* 564

26.1 I 95 caratteri ASCII stampabili 578
26.2 Alcune codifiche di ingresso per i caratteri latini 582

29.1 Gli accenti testuali e i simboli speciali 635
29.2 Numeri associati al livello di sezionamento 636
29.3 Parametri nella classe `book` per gestire gli oggetti flottanti . . . 679
29.4 Dichiarazioni per la scelta di famiglia, serie e forma 705
29.5 Corrispondenza fra comandi e dichiarazioni 705
29.6 Dichiarazioni di corpo 705

Elenco delle figure

1.1 Semplice esempio di composizione con LaTeX. 4

2.1 Dedica a Donald Knuth 11
2.2 Auguri di TUG (TeX Users Group) per il 2003 11
2.3 Confronto relativo fra le varie scale di lunghezza 12
2.4 Caratteristiche dei caratteri mobili 13
2.5 Cassa dei caratteri di una tipografia italiana 14

4.1 Schermata di LaTeXiT 47
4.2 Schema della tastiera italiana sulle piattaforme Windows 56
4.3 Schema della tastiera configurata con il driver EurKey 58
4.4 Schema della tastiera svizzera sulle piattaforme Windows 59
4.5 Schema della tastiera italiana su Linux senza personalizzazioni . 60
4.6 Schema della tastiera USA estesa sul MacBook Pro 61

6.1 Tre geometrie di pagina a confronto 119
6.2 Proporzioni ISO e aurea della pagina 121

7.1 Il comando `\centering` e l'ambiente *center* 145

9.1 Relazione fra le dimensioni della figura e l'origine degli assi . . . 179
9.2 Le potenzialità dell'ambiente *picture* secondo Lamport 183
9.3 Curve di Bézier di secondo e terzo grado nell'ambiente *picture* . 183
9.4 Moto uniformemente accelerato 184
9.5 Disegno ottenuto con il pacchetto *pgf* e l'ambiente *tikzpicture* . 186
9.6 Disegno nell'ambiente *tikzpicture* usando il pacchetto *circuitikz* 187
9.7 Una paginetta con l'angolo ripiegato 188
9.8 Un esempio di figura geometrica composta con METAPOST . . . 190
9.9 Istogramma con il risultato della valutazione di un seminario . . 201
9.10 Diagramma con il risultato della valutazione di un seminario . . 202
9.11 Diagramma a torta con i tipi di produzione dell'energia elettrica 203

10.1 Confronto sugli effetti dell'ingrandimento su due figure simili . . 213
10.2 Inclusione e conversione di un file `eps` 217

10.3 Due foto trattate con diverse chiavi 229

12.1 Le fasi per la produzione dell'indice analitico 243

13.1 Due esempi di strafalcioni giornalistici 266

15.1 Sette contro Tebe 301
15.2 Esempio di composizione in greco 303
15.3 Alcuni alfabeti gotici che includono anche i capilettera ornati . 306
15.4 Completamento del file sorgente 316
15.5 Risultato della composizione con XeLaTeX 317

16.1 Otto slide per una presentazione di cinque frame 326

18.1 Il carattere metallico della 'm' 344
18.2 Il corpo, l'interlinea e l'avanzamento di riga 344
18.3 Composizione normale e sterlineata 345
18.4 Legature antiche 347
18.5 Il simbolo di *maschio* fortemente ingrandito 351
18.6 Esempi di testi composti con i font Times e Palatino 360
18.7 Il Credo di Nicea in tre lingue che fanno uso di alfabeti diversi . 372
18.8 I *dingbats* creati da Hermann Zapf 376
18.9 Schema grafico del processo di composizione 393
18.10 Presentazione all'esterno dei risultati della composizione 393
18.11 Processi di conversione dei vari formati di uscita 394
18.12 Trasformazioni di formato dei font e generazione dei file ausiliari 395
18.13 Gestione delle mappe dei font 396

19.1 Alcune proprietà delle scatole 429

20.1 Imposizioni per diverse segnature 444
20.2 Esempio di crocini predisposti per una tipografia italiana 445
20.3 Disegno della pagina secondo Gutenberg 450
20.4 Disegno della pagina con il metodo delle strisce 451
20.5 Confronto fra tre gabbie auree 453

23.1 La finestra di compilazione di Scrivener 519
23.2 La compilazione da Scrivener 521
23.3 La compilazione da Scrivener in MMD o LaTeX 523

Avvertenze

In questo testo si parla di quello che si può fare con la ricca dotazione di programmi e di pacchetti presenti in ogni distribuzione del sistema TEX. Le distribuzioni del sistema TEX gratuite sono essenzialmente due, TEX Live e MiKTEX. In commercio ci sono diverse altre distribuzioni.

Qui ci si riferisce principalmente alla distribuzione del sistema eseguita mediante il DVD prodotto dalla Associazione Internazionale degli Utenti di TEX (TEX Users Group, TUG) e ridistribuito anche ai membri dell'Associazione Italiana, GUIT (Gruppo degli Utenti Italiani di TEX e LATEX). Volendo, il DVD può venire acquistato ad un costo nominale di US$ 50,00 dal sito `http://www.tug.org"/store/`, ma nel sito di TUG si dice che l'immagine ISO del disco può essere scaricata gratuitamente. L'immagine ISO va poi masterizzata su un DVD vergine e usata tranquillamente alla pari del DVD originale. La distribuzione scaricata dal sito TUG menzionato è completa e permette l'installazione sulle macchine Windows, Linux e Macintosh.

Anche la distribuzione MiKTEX (valida solo per le macchine Windows) può venire scaricata gratuitamente dal sito `http://miktex.org/2.9/setup`. Più precisamente da questo sito è possibile scaricare dei semplici programmi di installazione, uno molto piccolo per installare una versione di base e uno di dimensioni maggiori per scaricare la distribuzione completa. Benché la distribuzione di base possa essere arricchita in modo praticamente automatico via via che l'utente richiama funzionalità ancora non installate, si consiglia vivamente di eseguire l'installazione completa.

Nel seguito si farà riferimento specialmente alla distribuzione TEX Live, perché con minime varianti è identica per tutte le piattaforme di elaborazione più comuni. Anzi, anche per chi lavora su macchine Windows, sebbene esista l'eccellente distribuzione MiKTEX, espressamente prodotta per lavorare con i vari sistemi operativi della Microsoft, si consiglia di installare la versione TEX Live. Certe personalizzazioni necessarie per la composizione tipografica di alta qualità diventano relativamente difficili da fare con la distribuzione MiKTEX, prevista per utenti ai quali, lavorando con un sistema operativo "proprietario", non sono concesse troppe personalizzazioni.

Nel seguito quando si scriverà CTAN ci si riferirà al Comprehensive TEX Archive Network (la rete degli archivi completi del sistema TEX); questa rete è formata da tre server principali dislocati in varie parti del mondo e da un numero sterminato di "mirror" che ne rispecchiano il contenuto. I primi sono aggiornati in modo sincrono ogni notte; i secondi sono aggiornati in modo dilazionato dipendente dalla loro politica amministrativa, ma generalmente sono molto aggiornati anche i mirror.

Nel seguito quando si userà l'indicazione `$TEXMF` ci si riferirà alla cartella o folder che funziona da "radice" per gli "alberi" di cartelle del sistema TEX. Questo sistema, infatti, ha i suoi file collocati in molte cartelle gerarchicamente collegate fra di loro in strutture ad albero; la radice di ciascuno di questi alberi è posta nel disco fisso dell'elaboratore in posizioni diverse a seconda della macchina

e del sistema operativo. Ogni utente conosce la sua macchina e sa dove trovare le cartelle radice dei suoi programmi preferiti.

Tuttavia, tranne dove indicato in modo esplicito, qui si userà solo il segno di barra diritta '/' come separatore fra i nomi delle cartelle; questo è il separatore normale per le macchine che lavorano con i sistemi operativi di tipo UNIX, incluso il sistema operativo delle macchine Macintosh recenti, etichettato con la sigla Mac OS X. Nelle macchine Windows il separatore è notoriamente la barra rovescia '\'; ma anche questo, all'interno dei file che contengono i testi da comporre con i programmi del sistema TEX va sostituito con la barra normale, poiché la barra rovescia per il sistema TEX ha un significato speciale.

I lettori ci vorranno perdonare se abbiamo dato la preferenza sia a TEX Live, sia alla barra diritta; questo è anche legato al fatto che come il sistema TEX fa parte del software libero, così abbiamo dato la preferenza ai sistemi operativi liberi o quasi. Ovviamente non c'è niente di male a servirsi di prodotti commerciali, che talvolta offrono qualcosa di più dei prodotti liberi. Anche nel campo delle distribuzioni del sistema TEX ci sono diverse distribuzioni commerciali che offrono qualcosa in più. Tutti i prodotti commerciali, però, danno qualcosa in meno, nel senso che l'utente è vincolato al prodotto e deve talvolta rinunciare alla possibilità di trasferire agevolmente programmi e documenti da una macchina ad un'altra con un diverso sistema operativo o priva dello stesso prodotto commerciale.

Una certa conoscenza preliminare del sistema TEX è utile per comprendere a fondo questo testo; in ogni caso si dà per scontato che l'utente conosca bene come lavorare sulla propria macchina di elaborazione e abbia una certa familiarità con l'uso della finestra chiamata "Terminale" o "Prompt dei comandi" nei vari sistemi operativi. Talvolta il sistema TEX richiede necessariamente di dare dei comandi in linea, scritti cioè in una di queste finestre specificatamente destinate ad interagire con la macchina mediante comandi scritti, senza usare il mouse o altri comandi "scorciatoia".

Capitolo 1

Composizione sincrona e asincrona

1.1 Introduzione

Come si è detto i word processor sono programmi per la composizione di testi dove le singole parole sono collocate nel testo composto, direttamente visibili sullo schermo; idealmente quanto si vede sullo schermo dovrebbe andare anche in stampa tale e quale, senza modifiche di sorta.

Questo genere di programmi prevede una forte interattività fra compositore e programma di presentazione o di stampa; dato il fatto che ogni tasto produce subito un effetto, così come ogni movimento e click del mouse, il tipo di composizione che viene prodotto viene detto 'composizione sincrona', proprio perché ogni azione del compositore si traduce in una immediata variazione del testo composto.

Va da sé che, perché il programma sia davvero sincrono e il ritardo fra azione e visualizzazione sia trascurabile, la forza del programma deve essere concentrata nella rapidità della presentazione. Tale caratteristica non può che andare a scapito della 'perfezione' della composizione, perché questa dipende da una elaborazione molto più accurata sul testo da comporre. È vero che oggi i programmi di videocomposizione (word processor) sono estremamente rapidi e quindi ogni anno che passa la qualità della loro composizione migliora vistosamente, tuttavia il compromesso fra velocità e qualità esiste sempre.

La 'composizione asincrona' consiste invece nell'introdurre il testo da comporre in un file senza badare al suo aspetto grafico e nel farlo fluire successivamente dentro un programma di impaginazione, dove il compositore può agire (anche in un secondo tempo) per modificare interattivamente il solo aspetto grafico. Durante questo processo può accadere di dover modificare la giustezza di alcune linee o di interi capoversi e quindi può accadere che le linee facenti parte di un capoverso

o l'intero capoverso debbano venire ricomposti e giustificati diverse volte, senza limitarsi a ottimizzare la composizione della parte modificata, ma prendendo in esame l'intero capoverso; questo implica l'ottimizzazione compositiva dell'intero capoverso, non solo una aggiustatina del punto in cui si è eseguita la modifica. Ecco quindi che la composizione avviene in due tempi, l'introduzione del testo e l'ottimizzazione della composizione, avendo però a disposizione l'intero testo da trattare.

Va da sé che la composizione asincrona assicura una migliore qualità di composizione rispetto a quella sincrona, visto che non viene tenuta in nessun conto la velocità di visualizzazione, ma la forza compositiva viene concentrata sulla qualità.

1.2 Il mark-up di LaTeX

LaTeX appartiene al gruppo di programmi di composizione asincrona, ma con alcune particolarità; con un editor di testi l'autore o il compositore inserisce il testo in un file che viene successivamente elaborato con LaTeX che agisce da impaginatore; tuttavia il primo file prodotto mediante l'editor di testo non contiene solo il testo in senso stretto, ma contiene una serie di informazioni di mark-up che successivamente permettono a LaTeX di sapere che cosa sta componendo, in modo da eseguirne la composizione secondo le direttive dello stile del documento che sta elaborando.

Il testo prodotto mediante l'editor di testo è quindi una specie di programma, che contiene sia il testo da comporre, sia i comandi e le altre istruzioni necessarie per il riconoscimento della parte di testo in corso di elaborazione, sia le modalità compositive specifiche che il compositore desidera introdurre.

L'utente non deve spaventarsi con le parole 'linguaggio di programmazione'; per lo più si tratta solo di informazioni di mark-up per specificare titoli, testi speciali, composizione della matematica, e simili. I comandi e le altre specificazioni sono solitamente espresse in inglese (molto semplice, ridotto all'essenziale) evitando quasi sempre le abbreviazioni o gli acronimi.

Un breve esempio consente di capire meglio il genere di difficoltà, o meglio, di facilità di questo tipo di linguaggio di mark-up:

```
\section{Breve esempio}

Il  titolo precedente è quello di una sezione;
più propriamente la parola inglese \textit{section} in
tipografia indica ciò che in italiano viene chiamato
\textit{paragrafo}.

Questo secondo capoverso contiene una equazione numerata:
\begin{equation}
ax^2 + bx + c =0
```

```
\end{equation}
%
Esso contiene anche una tabella centrata:
\begin{center}
  \begin{tabular}{lcl}
    \hline
      Nome      & relazione  & parentela \\
    \hline
      Giovanni  & è          & il papà   \\
      Ada       & è          & la mamma  \\
      Maria     & è          & la figlia \\
      Giuseppe  & è          & il figlio \\
    \hline
  \end{tabular}
\end{center}
Qui c'è la fine del capoverso che contiene
sia una formula sia una tabella.
```

L'insieme di testo e comandi contenuti nell'esempio precedente viene poi composto come nella paginetta della figura 1.1.

Nel capitoli successivi verranno spiegate tutte le informazioni di mark-up e le istruzioni usate nell'esempio. Tuttavia anche il lettore con pochi rudimenti di inglese capisce perfettamente quello che il mark-up ha specificato; le istruzioni per la composizione del materiale tabulare sono un poco più articolate, ma guardando il risultato ci si rende conto che non sono poi così misteriose.

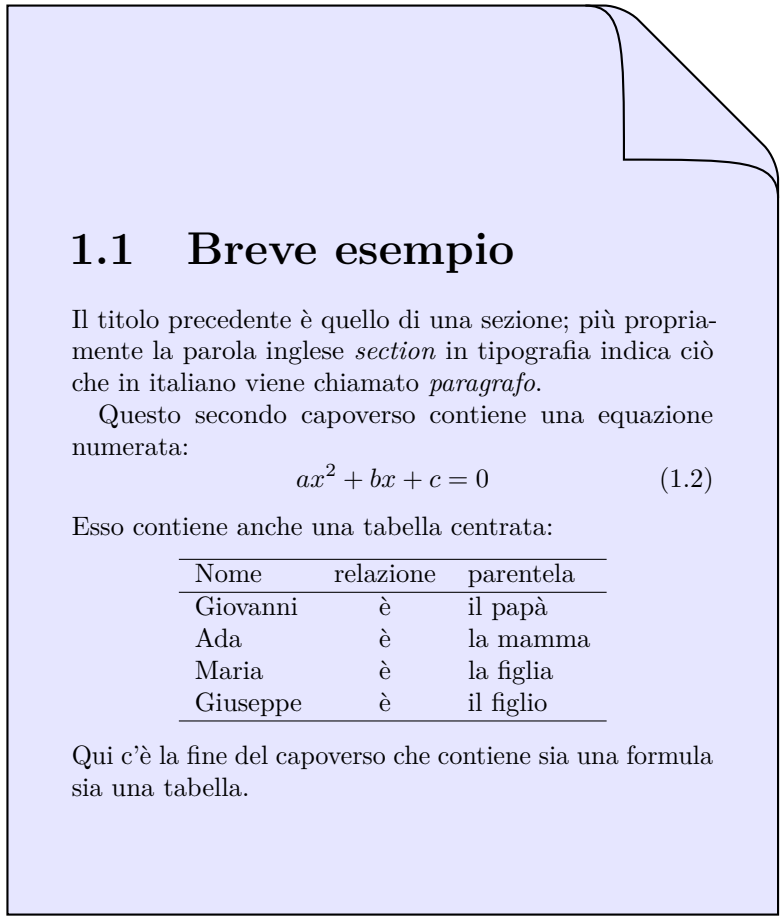
1.3 Scribus: un impaginatore libero con licenza GPL

Di programmi di impaginazione commerciali ne esistono molti e tutti, essendo destinati al lavoro professionale, sono anche abbastanza costosi. Tuttavia per le tre piattaforme di lavoro principali (Windows, Linux, Macintosh) esiste l'impaginatore Scribus che, come LaTeX, è rilasciato con licenza GPL e quindi è *Software Libero*, e chiunque se lo può scaricare dalla rete e installarselo.

Tuttavia merita riportare un piccola parte della documentazione descrittiva di Scribus; qui c'è una libera traduzione del documento che si trova in rete all'indirizzo `http://wiki.scribus.net/index.php/Word_Processing_vs_DTP`.

Prime indicazioni

Molti usano pacchetti di produzione per l'ufficio (*office suite*) come per esempio Microsoft Office™ oppure OpenOffice.org™. Ma a un certo punto non sono più soddisfatti delle potenzialità del loro word processor preferito. Forse non possono



1.1 Breve esempio

Il titolo precedente è quello di una sezione; più propriamente la parola inglese *section* in tipografia indica ciò che in italiano viene chiamato *paragrafo*.

Questo secondo capoverso contiene una equazione numerata:

$$ax^2 + bx + c = 0 \tag{1.2}$$

Esso contiene anche una tabella centrata:

Nome	relazione	parentela
Giovanni	è	il papà
Ada	è	la mamma
Maria	è	la figlia
Giuseppe	è	il figlio

Qui c'è la fine del capoverso che contiene sia una formula sia una tabella.

Figura 1.1: Semplice esempio di composizione con LaTeX.

apportare correzioni fini al layout, oppure alcune loro idee non sono realizzabili. Oppure la rivista all'edicola giù all'angolo appare fatta in modo più professionale.

Perciò queste persone si guardano attorno per trovare altri applicativi, e trovano applicativi professionali per il Desk Top Publishing (DTP) come QuarkXPress, InDesign o Scribus. L'ultimo in particolare ha un grande vantaggio: è Free Software, perciò perché non provarlo? Dopo aver scaricato il programma dalla rete, averlo installato e finalmente lanciato, l'autore abituato ai word processor resta seduto davanti allo schermo come "un coniglio di fronte a un serpente". Non sa assolutamente che cosa fare. Tutto appare così strano; e perché non può semplicemente introdurre il testo nel suo documento? Dopo qualche tempo, completamente frustrato, l'autore disinstalla Scribus e torna al suo solito word processor, accettandone le limitazioni e gli svantaggi.

Le principali differenze

- Un applicativo DTP non è fatto per creare testo — almeno se si sta pensando a grandi quantità di testo, cioè molte pagine. Per questo scopo è meglio usare un word processor o un semplice elaboratore di testi anche senza mark-up. Dopo che è stato creato un file di testo, esso può essere importato nell'applicativo DTP. Da questo momento all'interno dell'applicativo DTP si eseguono solo delle piccolissime modifiche al testo e si evitano le correzioni corpose usando lo stesso applicativo DTP. Questo modo di procedere separatamente deriva dalle diverse responsabilità dei professionisti: una persona crea il file di testo, un'altra persona ne crea il layout.

- La maggior parte degli applicativi DTP moderni sono basati sulle scatole (*frame*). Questo significa che tutto il contenuto di una pagina è collocata dentro a delle scatole. "Contenuto" include il testo, le figure, i disegni e ogni altra cosa che l'autore vuol vedere nella pagina stampata. Queste scatole sono spostabili con grande libertà e possono essere collocate in ogni punto della pagina. Il testo di una scatola può esser fatto proseguire in un'altra scatola.

  Si potrebbe obbiettare che anche i word processor hanno le scatole di testo e che è possibile importare una figura o un diagramma nel documento che si sta componendo con il word processor. Tuttavia appena si prova a mettere con precisione al suo posto l'oggetto, lo si vuole ridimensionare, aumentarne o diminuirne la risoluzione grafica, o ruotarlo, si scoprono rapidamente le grosse limitazioni di queste prestazioni "DTP–like". Spesso è difficile impedire che si spostino da sole quando si aggiunge altro materiale al documento.

- Una volta che il testo è nella sua scatola di testo, lo si può manipolare con un altissimo grado di precisione. Un word processor può mettere a disposizione corpi dei caratteri di 6, 7, 8, 9, 10... punti, ma con Scribus la precisione arriva ai decimi di punto. Perché è così importante? In questo modo si può far stare un dato testo dentro un'area di date dimensioni. Per esempio, potrebbe essere necessario adattare la larghezza di una parola con un elemento grafico decorativo che vi giace sotto.

  Si ha la possibilità di allungare un font sia orizzontalmente, sia verticalmente, aggiustare con precisione lo spazio fra le righe (ancora con una precisione di decimi di punto) e lo spazio fra le lettere (accostamenti, crenature, *kerning*). Gli applicativi DTP pertanto danno il controllo completo sulla precisa posizione dei caratteri all'interno del testo, e consentono di adattare il testo ad un dato spazio rettangolare o di qualunque altra forma. Questi sono esempi dove il controllo preciso sulle dimensioni e sulla posizione è assolutamente necessario. Si può adattare il testo in modo che segua il profilo di una figura (o anche il contrario — si provi ad adattare il

testo al profilo di una figura con un word processor — pur tuttavia questa potrebbe essere una specifica di layout del DTP), sovrapporre immagini al testo o viceversa, e creare le separazioni di colore per la stampa.

- Negli applicativi DTP la tavolozza dei colori contiene solo qualche dozzina di colori differenti, chi più, chi meno. Qualunque altro colore si desideri deve essere definito a mano. Questo perché nella stampa professionale in offset, i colori aggiunti possono essere considerati *spot color* e la cosa potrebbe non essere quello che si desidera.

  Il DTP consente di rendere semitrasparenti le immagini, i grafici, il testo — qualunque cosa si possa mettere in una pagina.

- Si possono definire le cosiddette *master page*. Se una pagina normale usa una master page come base, tutti gli elementi già presenti sulla master page appaiono anche nella pagina normale. Si pensi ad una rivista, a una *newsletter*, dove certi elementi, come per esempio le intestazioni, i piedini, i numeri di pagina, i loghi, e il layout della pagina sono ripetitivi, mantenendo però una differenza fra le pagine di sinistra o quelle di destra. All'interno dello stesso documento può essere definita più di una master page; un documento di molte pagine potrebbe fare uso di un numero considerevole di master page.

Che cosa un applicativo DTP non può fare

- Un applicativo DTP serve per creare il layout di un documento; non serve per strutturare un testo. Non può creare riassunti o indici al posto del compositore. Non ci sono funzioni per aggiungere e/o gestire note in calce. Scribus 1.3.x, a differenza di altri DTP, può generare l'indice.

- In generale è difficile connettere un applicativo DTP a dati esterni, come quelli contenuti in un database. Scribus lo può fare mediante un suo plug-in python destinato a ciò.

Aggiungo, su informazione di Gianluca Pignalberi, che Scribus è in grado di fareelaborare al volo dati da programmi esterni e poi di importarli; fra questi programmi esterni ci sono LaTeX, LilyPond (testi musicali), Gnuplot (diagrammi bi- e tridimensionali) POV-Ray (tracciamento di raggi) e Graphviz (tracciamento di grafi).

Vantaggi degli applicativi DTP

- Controllo assoluto del layout. Gli applicativi DTP non rimescolano mai le scatole per aggiustarle alle necessità del testo o della stampante. Al contrario si devono impostare le specifiche del testo e/o della stampante in modo corretto.

- Il profilo del testo si può adattare a qualunque profilo irregolare.
- Le griglie, i margini e i righelli sono molto utili per collocare correttamente le scatole.
- L'uscita è professionale. I formati PDF o PostScript, o uno di quei formati necessari per quelle enormi macchine di stampa offset? Con un applicativo DTP li si può ottenere tutti.

1.4 Conclusione

Se il lettore di queste note desidera disporre di un potente mezzo per creare dépliant pubblicitari, gli strumenti DTP gli sono indispensabili.

Ma se il lettore vuole produrre testi scritti e illustrati può constatare che LaTeX fa tutto o quasi tutto quello che può fare un applicativo DTP, anche adattare il profilo di un testo a quello di una immagine. Il posizionamento di ogni elemento può essere fatto non con la precisione di un decimo di punto, ma con quella di $1/2^{16}$ di punto. Normalmente non è necessario gestire scatole di testo, ma lo si può fare e in questo testo c'è anche un esempio. Le figure e i grafici sono sempre gestiti come scatole. Il colore può essere gestito sia con una tavolozza ridotta, sia con collezioni di colori abbastanza ampie attraverso un apposito plug-in, il quale consente anche di gestire la trasparenza degli oggetti colorati, testo incluso.

L'uscita può essere di default in formato PDF. La variazione della larghezza e/o dell'altezza del corpo dei caratteri viene gestita automaticamente per la composizione del testo (microgiustificazione), ma LaTeX e i suoi plug-in (estensioni) contengono comandi per modificare, ruotare, e scalare qualunque oggetto; il risultato è ottimale se l'oggetto ha carattere vettoriale.

Di default LaTeX lavora con quattro master page che distinguono le pagine di sinistra da quelle di destra. Ma mette a disposizione una vasta serie di comandi per creare e poi usare quante master page si vogliano.

LaTeX genera gli indici generali e analitici; genera i glossari; è in grado di produrre diversi livelli di apparati critici nei testi letterari, permette di comporre la musica, permette di comporre con qualunque alfabeto e con qualunque tipo di font, anche retrogrado o ad ideogrammi; con i font *bitmapped* e con quelli scalabili, anche se, ovviamente, ottiene risultati migliori con questi ultimi; produce correttamente i riferimenti incrociati e numera automaticamente tutti gli oggetti numerabili, quali i paragrafi, le figure, le tabelle, le formule, eccetera; è in grado di includere praticamente qualunque immagine in qualunque formato (be', talvolta con un po' di trasformazioni preliminari) e di produrre disegni al tratto o con sfumature di colore che hanno il pregio di venire composti con le parti testuali formate con gli stessi font usati per il testo. Inoltre, ultimo, ma non meno importante, LaTeX non ricorre a formati di file proprietari al fine di poter mantenere l'interoperabilità con i calcolatori più diversi. LaTeX insomma fa certamente di più di un qualsiasi impaginatore, anche se ha un 'peccato

originale', secondo gli utenti che non ne apprezzano abbastanza il valore, cioè: *quello che LaTeX non può fare, essendo un programma di composizione asincrona, è quello di consentire al compositore di vedere interattivamente il risultato dei suoi comandi*, e questo, come spiegato in precedenza, ha molti vantaggi e alcuni piccoli svantaggi.

Capitolo 2

Nozioni elementari di tipografia

Il gergo della tipografia, come tutti i gerghi professionali, contiene un certo numero di parole che hanno un significato che deriva dalla pratica o dalla tradizione.

2.1 Tipografia e dattilografia

Bisogna innanzi tutto chiarire che comporre tipograficamente è una cosa del tutto diversa dalla composizione con la macchina da scrivere e perciò anche con i moderni sostituti delle macchine da scrivere, costituiti dai word processor.

Questi ultimi oggi sono talmente avanzati che danno l'illusione di poter eseguire una composizione tipografica; ma in realtà, a parte i compromessi fra velocità di presentazione e perfezione della composizione illustrati nel capitolo precedente, il risultato dipende moltissimo dalla professionalità del "tastierista"; costui o costei è la persona che immette il testo e che decide cosa fare, che spazi lasciare, dove mettere il materiale non testuale (per esempio le fotografie) eccetera. Se l'operatore, designato con il vecchio termine di tastierista, non ha abbastanza professionalità, il risultato della composizione è modesto.

La bellezza di una composizione tipografica, a parte il testo, sta nel fatto che la disposizione del materiale da leggere o da consultare non richiama su di sé l'attenzione, ma mantiene quella sobrietà che consente al lettore di recepire il messaggio senza distrazioni inutili e senza zoppicare nel leggere a causa di spaziature irregolari o continui cambiamenti di stile dei caratteri.

Vale la pena di osservare la figura 2.1 dove Herman Zapf, un grande disegnatore di caratteri contemporaneo, usa il suo font Zapfino, destinato proprio alle citazioni o alle epigrafi, per comporre una frase di Oswald Veblen che esalta la matematica (e il suo linguaggio), proprio il motivo che spinse Knuth a "inventare" la tipografia assistita da calcolatore mediante il programma TEX. Una

analoga motivazione celebrativa sta alla base del biglietto di auguri del 2003 della figura 2.2 che l'associazione internazionale degli utenti di TEX ha inviato ai suoi soci.

È chiaro che questa epigrafe e questo biglietto richiamano l'attenzione su di sé; sono fatti apposta! Ma un libro completamente composto con il font Zapfino sarebbe faticosissimo da leggere e il lettore sarebbe continuamente distratto dalle acrobazie calligrafiche eseguibili con questo font[1].

La scelta dei font, la loro grandezza, il loro stile, la loro nerezza, la scelta della distanza fra le righe, i margini, gli spazi bianchi lasciati attorno agli oggetti non testuali, sono tutte cose di competenza del "disegnatore editoriale", non di dilettanti come siamo tutti noi. LATEX, nei limiti di quello che può fare un oggetto "inanimato" come un programma di elaborazione di dati, sostituisce sia il disegnatore editoriale sia il tipografo; non sostituisce il tastierista; se questo pensa di essere davanti ad una macchina da scrivere un po' sofisticata, nonostante tutto quello che LATEX è capace di fare, la composizione risulterà squilibrata e disomogenea, in sostanza, l'opera di un dilettante.

La raccomandazione, perciò è la seguente: lasciamo LATEX fare il suo mestiere! Il risultato sarà sicuramente migliore di quello che potremmo ottenere con i nostri maldestri interventi "correttivi".

2.2 Unità di misura tipografiche

In tipografia si usano molte unità di misura tipografiche che spesso, se non sempre, non hanno nulla a che fare con il sistema metrico decimale. Essenzialmente tutto viene misurato in punti tipografici o in loro multipli.

Qui nasce una prima difficoltà interpretativa. Che cosa è un punto tipografico? Ne esistono almeno tre versioni:

1. Il punto tipografico anglosassone definito come la frazione 1/72,27 di un pollice, a sua volta pari a 25,4 mm. Ne segue che il punto anglosassone corrisponde a 0,3514598 mm, poco più di un terzo di millimetro. Questa è l'unità di misura usata da LATEX, e ogni informazione metrica che LATEX fornisce al compositore durante la sua esecuzione è sempre espressa in questo tipo di punti tipografici.

2. Il punto PostScript corrisponde a 1/72 di pollice, quindi è un poco più grande del punto anglosassone perché vale 0,35277778 mm. La differenza sembra piccola, ma quando si parla di diverse decine di punti la differenza è visibile a occhio nudo.

[1] Recentemente chi scrive ha trovato il libretto di Lorenza Bonifazi, *... e si mandi a tavola – Antiche ricette del Montefeltro*, Raffaelli Editore, Rimini 2006. Il libretto ha la parte contenente le ricette composta con il font Zapfino e per "imparare" a leggerlo ci vuole un pochino. Lo scopo dichiarato di usare questo font calligrafico molto ornato è quello di rendere l'atmosfera originale del quaderno dal quale l'autrice ha ricavato le ricette; il quaderno era stato scritto a mano, con bella calligrafia, da una sua antenata a metà del XIX secolo.

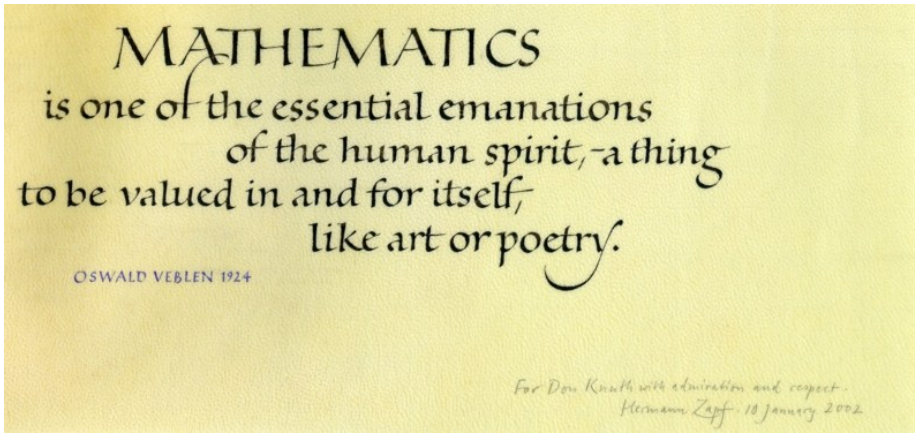

MATHEMATICS
is one of the essential emanations
of the human spirit, a thing
to be valued in and for itself,
like art or poetry.
OSWALD VEBLEN 1924
For Don Knuth with admiration and respect.
Hermann Zapf. 18 January 2002

Figura 2.1: Dedica a Donald Knuth

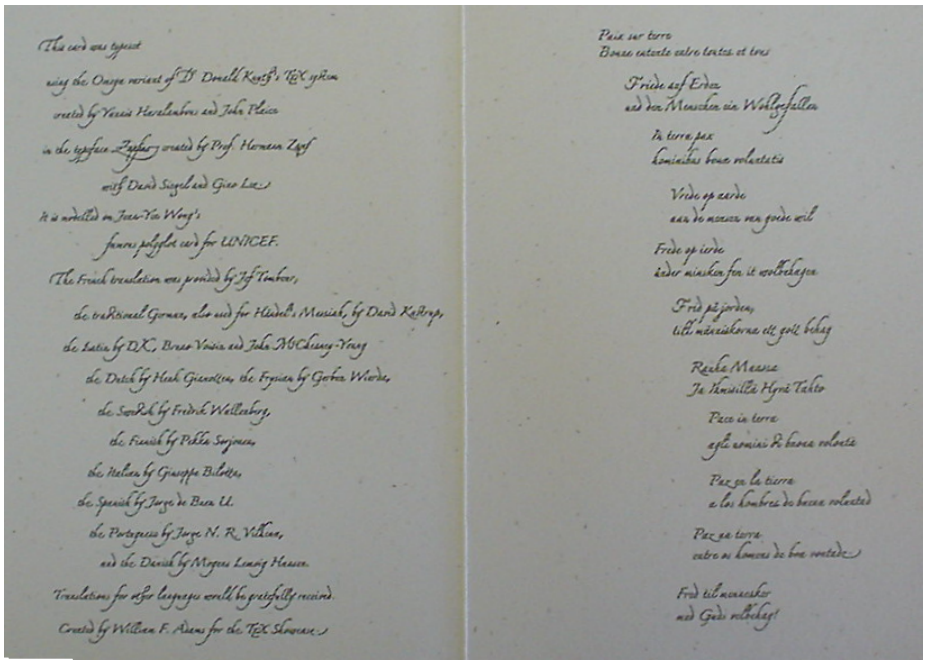

This card was typeset
using the Omega version of D. Donald Knuth's TEX system
created by Yannis Haralambous and John Plaice
in the typeface Zapfino created by Prof. Hermann Zapf
with David Siegel and Gino Lee
It is installed on Jean-Yves Wong's
famous polyglot card for UNICEF.
The French translation was provided by Jef Tinbar,
the traditional German, also used for Händel's Messiah, by David Kastrup,
the Latin by D.K., Bruno Voisin and John McCleary-Yang
the Dutch by Hank Gieselman, the Frysian by Gosse Wierda,
the Swedish by Friedrich Wollenberg,
the Finnish by Pekka Sarjanen,
the Italian by Giuseppe Bilotta,
the Spanish by Jorge de Buen U.
the Portuguese by Jorge N. R. Villaça,
and the Danish by Mogens Lemvig Hansen.
Translations for other languages would be gratefully received.
Created by William F. Adams for the TEX Showcase
Pace sur terre
Bonne entente entre toutes et tous
Friede auf Erden
und den Menschen ein Wohlgefallen
In terra pax
hominibus bonae voluntatis
Vrede op aarde
aan de mensen een goede wil
Fride op jorde
åt människorna ett gott behag
Rauhe Meerie
Ja Hyvänsä Hyvä Tahto
Pace in terra
agli uomini di buona volontà
Paz en la tierra
a los hombres de buena voluntad
Paz na terra
entre os homens de boa vontade
Fred til mennesker
med Gudi velbehag!

Figura 2.2: Auguri di TUG (TEX Users Group) per il 2003

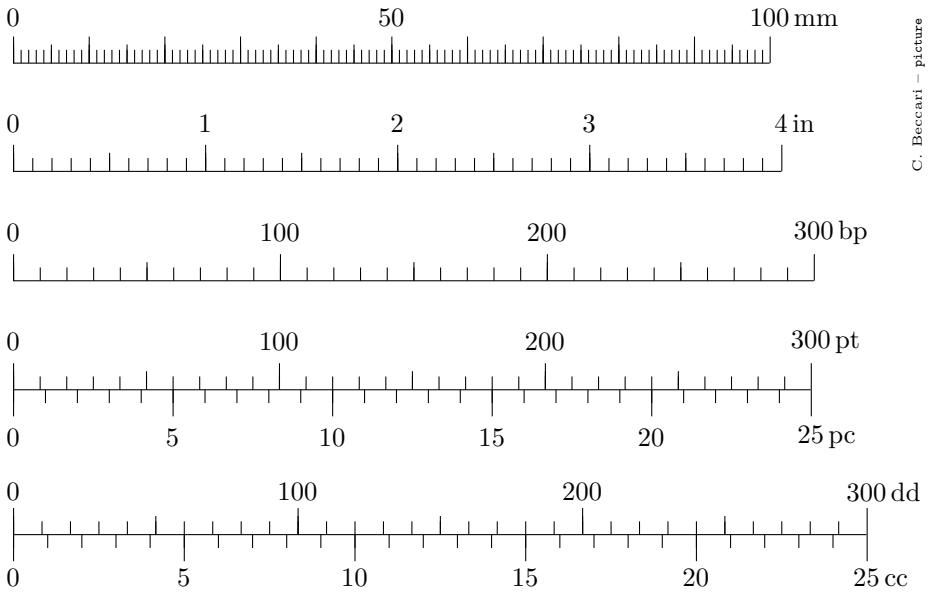


Figura 2.3: Confronto relativo fra le varie scale di lunghezza

3. Il punto didot viene usato nell'Europa continentale; esso corrisponde a 0,3760 mm e basta una decina di punti per vedere la differenza ad occhio nudo rispetto al punto anglosassone.

La figura 2.3 presenta un confronto (relativo) delle varie scale metriche che si possono ottenere con le unità di misura descritte in questo paragrafo; se la pagina viene stampata in scala 1 : 1, allora il confronto diventa anche un confronto assoluto.

Non è il caso di preoccuparsi di queste differenze; tuttavia nasce spontanea la domanda: "Ma perché i tipografi non si sono accordati su una unità metrica?" Non so se questa risposta sia adatta: penso che la tipografia sia un'arte praticata in un ambito piuttosto ristretto dove gli interscambi erano fino alla fine dell' '800 abbastanza scarsi, quindi la possibilità di fraintendimenti era modesto. Oggi, nel mondo dell'informatica e della tipografia assistita da calcolatore, sarebbe desiderabile una maggiore uniformità. Infatti oggi si sta imponendo in tipografia il punto anglosassone e in quella assistita dal calcolatore il punto PostScript. È un peccato che il Sistema Internazionale non sia riuscito nel compito di standardizzare una unità di misura adatta alla tipografia (per esempio 'un terzo di millimetro', ma dotato di tutti i multipli e sottomultipli decimali del caso) come invece, in parte, è riuscito a fare con il carato metrico.

Per indicare lunghezze maggiori in tipografia si usano multipli duodecimali, come il pica, che rappresenta 12 punti anglosassoni, o il cicero che rappresenta 12 punti didot. Bisogna però dire che mentre nel Nord America si usa il pica

Figura 2.4: Caratteristiche dei caratteri mobili. (*a*) Visione d'insieme di un carattere mobile metallico; (*b*) vista laterale del fusto del carattere; (*c*) vista della faccia con il disegno rovesciato del segno da stampare.

come unità di misura per indicare la giustezza (lunghezza) della riga di stampa, in Europa e in molte altre parti del mondo più o meno 'decimalizzate', Regno Unito compreso, la giustezza si indica in millimetri. Lo stesso vale per i formati delle carte da stampa; nel Regno Unito sopravvivono denominazioni verbali per i formati dei libri, ma poi la carta è misurata in millimetri. In Italia il cicero è spesso indicato col nome di 'riga', salvo poi a non essere più chiaro se si parla di una riga di testo o di una riga alta 12 punti e, ai giorni nostri, se i 12 punti siano didot o anglosassoni.

2.3 Misure tipografiche

La grandezza dei caratteri è indicata in punti; quando si lavorava con i caratteri mobili metallici, era chiarissimo che cosa volesse dire la parola 'corpo' dei caratteri; questi infatti erano ricavati da blocchetti parallelepipedi di metallo in cui una delle lunghezze rappresentava l'altezza del blocchetto; su una delle facce di base era ricavato in rilievo il disegno del segno da stampare. Si vedano le indicazioni costruite nella figura 2.4, dove sono evidenziate le grandezze dei caratteri mobili.

La faccia di base aveva una altezza e una larghezza. La larghezza era variabile a seconda del segno che il blocchetto conteneva; è evidente che il blocchetto di una 'M' era molto più largo di quello di una 'i'. Invece l'altezza era uguale per tutti i blocchetti dello stesso alfabeto, o meglio dei blocchetti della stessa 'cassa'; la cassa era un cassetto diviso in tanti scomparti quanti erano i segni minuscoli, maiuscoli, segni di interpunzione, cifre, eccetera, che formavano la collezione dei caratteri dello stesso 'disegno'; la figura 2.5 mostra la disposizione di una cassa italiana. Bene, l'altezza della base di tutti i blocchetti della stessa cassa formava il corpo e veniva misurato in punti.

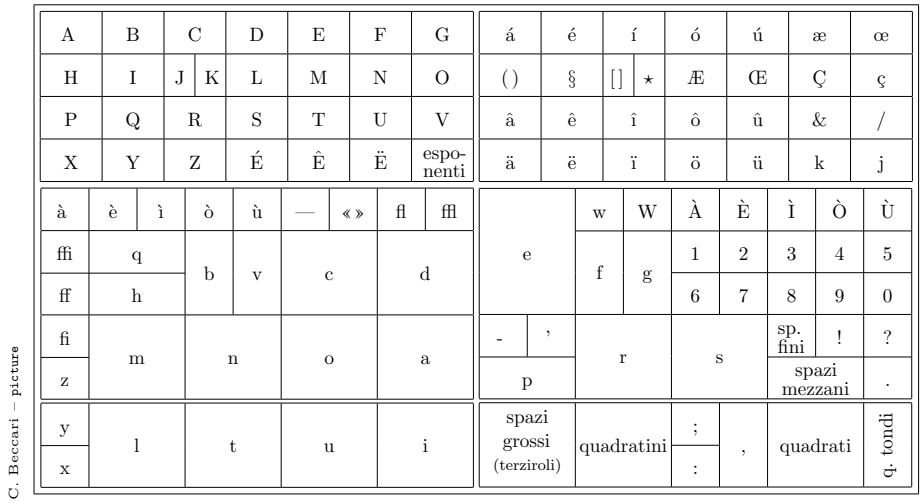

A B C D E F G
H I J K L M N O
P Q R S T U V
X Y Z É Ê Ë esponenti
á é í ó ú æ œ
( ) § [ ] ⋆ Æ Œ Ç ç
â ê î ô û & /
ä ë ï ö ü k j
à è ì ò ù — « » fl ffl
ffi q b v c d
ff h
fi m n o a
z
y l t u i
x
e w W À È Ì Ò Ù
f g 1 2 3 4 5
6 7 8 9 0
- , r s sp. fini ! ?
p spazi mezzani .
spazi grossi (terziroli) quadratini ; , quadrati q. tondi
:
C. Beccari – picture

Figura 2.5: Disposizione delle lettere e degli altri segni negli scomparti di una cassa usata in una tipografia italiana

All'indirizzo `http://www.youtube.com/user/EleonoraTallone#p/u/` si trova un breve filmato realizzato da Eleonora Tallone che "intervista" Enrico Tallone, un tipografo artista che compone a mano con caratteri mobili e pubblica testi di grande pregio artistico tipografico. Nel filmato si vedono chiaramente non solo le casse, i caratteri mobili, il compositoio, ma anche altri strumenti tipografici e almeno una macchina da stampa, una platina, in funzione. Un breve filmato semplice ed istruttivo.

Quindi un carattere di 12 pt era un segno qualunque della cassa il cui blocchetto aveva una altezza della faccia di base di 12 punti. Questo evidentemente non rappresentava l'altezza del carattere, ma, ripeto, l'altezza della faccia del blocchetto. Il fatto che i blocchetti avessero le facce di base della stessa altezza, permetteva al compositore di adagiare tutti quelli di una riga sul compositoio e poi di trasferire la riga completa sulla pagina parzialmente composta con la certezza che tutti i caratteri della riga fossero allineati correttamente.

Inoltre il fatto che tutti i blocchetti avessero la stessa altezza, come la si vede rappresentata nella figura 2.4(b), assicurava che essi venissero inchiostrati omogeneamente con i rulli inchiostratori e poi prendessero simultaneamente contatto con la carta da stampare e ricevessero la stessa pressione dal torchio di stampa, così da imprimere i segni inchiostrati nella stessa maniera al fine di assicurare una nerezza omogenea dello stampato.

Oggi che i caratteri mobili metallici non si usano quasi più, che significato ha il corpo? Indica solamente l'altezza complessiva della riga di testo, spazio bianco incluso; e in pratica la distanza fra le righe di base di due righe successive quando queste non siano interlineate (nemmeno sterlineate; per questo concetto

si veda il capitolo 18).

Ecco allora alcuni nuovi termini: *linea di base*, *interlinea* e *avanzamento di riga*.

La linea di base è quella linea ideale sopra la quale sembrano appoggiati tutti i segni di una riga; sotto alla riga di base sporgono le parti discendenti dei segni come quelli delle lettere g, p, q, y; sopra la linea di base sporgono tutte le lettere, ma alcune come b, d, f, eccetera, sono più alte delle altre lettere minuscole, come le lettere a, c, e, m, n, x, eccetera. Siccome tutte queste minuscole sono alte come una lettera 'x', il loro 'occhio' (la loro altezza) in LaTeX viene indicato con il nome di *x-height*, cioè 'altezza della x'.

I *discendenti* sono i tratti delle lettere che sporgono sotto alla linea di base e determinano la *profondità* della riga. Gli *ascendenti* sono i tratti delle lettere minuscole che sporgono sopra la linea tangente alla 'x'. Il più alto ascendente è spesso quello dalla lettera 'h'; talvolta sporge sopra all'altezza delle maiuscole, talvolta è alto quanto le maiuscole. In ogni caso in ogni riga la sua altezza è determinata dall'ascendente più alto. Non necessariamente la somma della profondità e dell'altezza di una riga di testo è uguale al corpo.

Come già detto, di fatto il corpo è la distanza di due righe di base composte senza inserire alcuna interlinea. Poiché non necessariamente una data linea contiene tutti i caratteri con i discendenti e tutti quelli con gli ascendenti, il corpo può sembrare maggiore dell'altezza complessiva di una riga, ma in generale, anche se i segni di due righe successive composte senza interlinea non interferiscono fra di loro, la mancanza di interlinea spesso rende il blocco di testo troppo compatto e faticoso da leggere.

Per questo motivo si inserisce una interlinea di un paio di punti, o proporzionalmente di più per i corpi più grandi. Tradizionalmente l'interlinea era quella striscia di metallo che veniva inserita fra due righe successive per distanziarle un poco; in inglese sia chiama *leading* o *lead*, da pronunciare come 'lead' (piombo), perché queste strisce metalliche erano di piombo (o della stessa lega di piombo usata per i caratteri da stampa).

Di conseguenza l'*avanzamento di riga* rappresenta la distanza effettiva fra le linee di base di due righe successive composte interponendo l'interlinea. Si dirà quindi che un certo testo è composto in corpo 12/14 per dire che i caratteri hanno un corpo di 12 punti e che l'avanzamento di riga è di 14 punti, il che implica la presenza di una interlinea di 2 punti. A questo proposito si veda la situazione appena descritta rappresentata nella figura 18.2.

2.4 Le particolarità dei caratteri

Ognuno avrà notato che i caratteri da stampa si dividono sostanzialmente in due grosse categorie, ma in realtà essi sono distinti da una varietà di caratteristiche che permettono moltissime diverse classificazioni.

La differenza fra i caratteri usati dalle vecchie macchine da scrivere, a spaziatura fissa, e i caratteri da stampa vera e propria, a spaziatura variabile, credo che

sia evidente a tutti. Oggi i caratteri a spaziatura fissa sono quasi del tutto scomparsi, ma trovano importanti applicazioni, specialmente di carattere informatico, e non solo, quando è necessario evidenziare con il tipo di segno usato certi scritti tecnici, come i brani di linguaggio di programmazione, che non devono essere giustificati e non devono avere nomi o parole chiave divisi in sillabe in fin di riga, ma la cui spaziatura fissa aiuta il lettore a distinguere le variabili dagli operatori, quindi a leggere il codice con più facilità.

I caratteri a spaziatura variabile sono più comodi da usare nei testi giustificati e consentono in generale una lettura più agevole, specialmente se sono caratteri con grazie. Le grazie sono quei piccoli segni, solitamente con andamento orizzontale, alle estremità delle aste che rendono maggiormente definito il contorno dei singoli segni e, specialmente le grazie inferiori, consentono di allineare meglio la direzione della lettura lungo la riga di testo. Quindi una seconda classificazione riguarda la presenza di grazie (*serif* in inglese) o la loro assenza (*sanserif*).

I caratteri non hanno poi la stessa forma; ognuno è in grado di distinguere la forma dei caratteri tondi (*roman* in inglese) dai caratteri *inclinati* (*slanted* in inglese); questi a loro volta si distinguono dal fatto che sono ottenuti dal tondo semplicemente inclinando le aste, oppure sono disegnati apposta con andamento più corsivo (*italic* in inglese), sebbene di questo corsivo sia possibile disporre anche di una versione non inclinata.

Inoltre un alfabeto può contenere sia le lettere maiuscole, sia quelle minuscole dalla forma completamente distinta da quella delle maiuscole, oppure può contenere un alfabeto minuscolo in cui le lettere minuscole assomigliano ad una versione ridotta delle maiuscole; in quest'ultimo caso si è in presenza del maiuscoletto (*small caps* in inglese).

La nerezza dei segni può assumere livelli diversi; le collezioni di caratteri più ricche hanno diverse gradazioni di nero: chiarissimo, chiaro, normale, neretto, nero, nerissimo. I font per tipografia assistita dal calcolatore spessissimo hanno solo una versione normale e una nera, ma le collezioni più ricche possono avere diverse gradazioni.

Per altro gli stessi segni, con il medesimo grado di nero, possono avere dei disegni ristretti (*condensed*) o allargati (*extended*) rispetto alla dimensione normale.

Una caratteristica che invece non è visibile è la codifica; questo è un concetto tipicamente informatico, nel senso che ogni segno si trova descritto dentro un file insieme agli altri segni dello stesso alfabeto, e per recuperare quel segno dal file è necessario conoscere il suo 'indirizzo'. Esistono moltissimi modi di indirizzare i caratteri (i singoli segni) all'interno di un file e la cosa dipende anche dal numero di segni che il file contiene. Oggi la codifica UNICODE permette di avere file enormi che contengono un numero enorme di segni, disegnati in modo più o meno uniforme anche se appartengono ad alfabeti diversi. Non è raro il caso di file relativi ad una sola collezione che contengono l'alfabeto latino con i suoi segni numerici e di interpunzione, l'alfabeto greco; l'alfabeto cirillico, l'alfabeto ebraico, l'alfabeto arabo, una certa collezione di segni cinesi, giapponesi, coreani, raccolte di segni speciali, come i simboli matematici o i segni astrologici, e chi

più ne ha più ne metta, e non è difficile arrivare a qualche decina di migliaia di segni.

La codifica è importantissima, perché si scrive assumendo una certa codifica e si vede sullo schermo il testo correttamente rappresentato; ma poi se si cambia font, si rischia di non avere più la possibilità di decifrare il messaggio; a me è capitato più di una volta di aprire con Word dei documenti composti con un'altra versione di Word e trovare che i segni originali erano stati sostituiti dai segni matematici; la lettura evidentemente era diventata impossibile.

La cosa più importante per i professionisti che si occupano della grafica editoriale consiste nella scelta dei font; due font di forma tonda diritta, nerezza media, estensione normale, con lo stesso corpo e relativi allo stesso alfabeto possono apparire completamente diversi a seconda del rapporto fra lo spessore delle aste spesse e quelle sottili, a seconda della forma delle grazie, a seconda dell'inclinazione dell'asse ottico (non l'inclinazione delle aste), può apparire completamente diverso e può dare luogo ad una lettura più facile oppure più faticosa, se composta in righe con lo stesso avanzamento e della stessa lunghezza.

Questo testo è composto con i caratteri della collezione Computer Modern, con codifica a 128 caratteri del sistema TEX[2]; per lo più viene usata la forma di nerezza media e larghezza normale; il corpo normalmente usato è il corpo 10 composto con un avanzamento di riga di 12 punti. Talvolta si usa il corsivo per mettere in evidenza delle parole o per scrivere quelle in inglese. I titolini dei paragrafi sono scritti in tondo diritto, nerissimo esteso in corpo 14,4 con un avanzamento di 20 punti; l'avanzamento dei titolini in realtà non è molto importante perché quasi tutti si svolgono su una sola riga.

2.5 I contrografismi

Logicamente la pagina stampata contiene i segni che convogliano il messaggio, che vengono collettivamente denominati *grafismi*. I *contrografismi* sono le parti della pagina prive di segni, eventualmente sono solo colorati, ma svolgono una funzione essenziale nel disegno grafico della pagina, specialmente nella qualità della composizione, la cui lettura viene agevolata dal loro uso giudizioso.

Non solo l'interlinea è un contrografismo, ma anche la rientranza dei capoversi, gli spazi sopra e sotto alle figure, alle formule, ai titolini; i quattro margini di ciascuna pagina, quello superiore, quello esterno, quello inferiore e quello interno.

Questi ultimi, infatti, determinano la posizione della gabbia che contiene il testo e talvolta anche la *testatina* e/o il *piedino* (l'intestazione della pagina e la

[2]Inizialmente si era usata la collezione European Computer Modern con codifica a 256 caratteri, ma poi si è ripiegato sulla "vecchia" collezione con codifica a 128 caratteri, perché alcune delle persone che hanno collaborato alla stesura si sono trovate con una installazione del sistema TEX... troppo personalizzata, al punto da non essere più compatibile con quella degli altri collaboratori. Sono cose che succedono quando si usa uno strumento troppo potente. Dopo diverse versioni di questo testo si è deciso infine di usare la collezione Latin Modern a 256 caratteri, lasciando al singolo collaboratore la libertà di usare questi font, o di escluderli, ritornando perciò all'uso dei Computer Modern.

riga di piè di pagina; *header* e *footer* in inglese); se questi due elementi sono vuoti o quasi vuoti essi contribuiscono più alla grandezza dei margini che alla grandezza della gabbia.

I margini, specialmente quelli esterni, secondo un disegno grafico abbastanza tradizionale, quando sono sufficientemente ampi possono accogliere le note marginali; quindi le note possono trovare posto sia in calce alla pagina, sia nei margini; in qualche scritto di stile critico–letterario esse possono essere anche collocate in fondo a ogni capitolo.

Ma i margini possono accogliere anche parte delle figure o le didascalie delle figure; non si tratta quindi di spazio 'sprecato' ma di spazio che oltre a dare un tono alla pagina può svolgere funzioni ausiliarie al testo in modo creativo e spesso gradevole.

2.6 Le parti di alcuni documenti a stampa

Un libro, generalmente è la forma di stampato più complessa e a questa mi riferirò nel descrivere le parti.

Un rapporto è uno stampato che può assumere la forma di un libretto, ma generalmente ha meno pretese stilistiche, sia per la relativa brevità sia per il contenuto spesso assai tecnico (il che non vuol dire solo ingegneristico, ma anche giuridico, economico, eccetera).

L'articolo è invece un tipo di stampato, di solito assai breve e molto conciso; la concisione implica che non viene 'sprecata' un'intera pagina per presentare il titolo; spesso l'articolo è scritto su due colonne, il che è un artificio per usare gabbie di testo più larghe, capaci di contenere colonne con righe relativamente corte e facili da leggere, mentre se fosse composto ad una sola colonna in una gabbia così larga le righe risulterebbero troppo lunghe e quindi faticose da leggere.

Un libro può essere confezionato *incassato* (con copertina rigida) o *brossurato* (con copertina morbida incollata); all'interno della copertina esso comincia con un foglio generalmente privo di ogni scritto, seguito da una pagina, sulla cui facciata di destra (il *recto*) può essere riportato l'*occhiello*; il retro (il *verso*) è sempre senza testo; la pagina successiva sul recto presenta il titolo, mentre sul verso presenta le informazioni di carattere legale richieste, appunto, dalla legge.

La successiva pagina può contenere una dedica sul recto, mentre sul verso è senza testo; se la dedica manca, manca l'intero foglio.

Segue poi sul recto del foglio successivo una sezione non numerata generalmente intitolata 'Presentazione' o con un titolo equivalente. Spesso non supera una pagina, ma può anche occupare qualche pagina.

La successiva sezione, che si apre sul recto, contiene l'indice generale a cui possono seguire, aprendosi sul recto delle pagine, l'elenco delle figure, l'elenco delle tavole, l'elenco delle illustrazioni fuori testo; ovviamente la presenza di questi elenchi dipende dal tipo di libro, dal suo contenuto. Nei libri di carattere letterario, come i romanzi, spesso l'indice si trova in fondo al libro e in Italia era

questa la consuetudine di ogni tipo di libro molte decine di anni fa. Oggi molti romanzi e tutti i libri con indici molto strutturati presentano l'indice nelle prime pagine, come descritto sopra.

L'Introduzione segue queste parti iniziali, ma in Italia il materiale iniziale, *front matter* in inglese, termina solitamente prima dell'Introduzione. Le pagine che contengono il materiale iniziale sono generalmente numerate con numeri romani; terminato il materiale iniziale, la numerazione delle pagine ricomincia da 1 e viene scritta con cifre arabe fino alla fine del libro[3].

Dopo il materiale iniziale comincia il corpo del testo, *main matter* in inglese.

Il corpo del libro è spesso diviso in parti, ma è sempre diviso in capitoli; ogni capitolo è spesso diviso in sezioni gerarchiche che in italiano si chiamano paragrafo, sottoparagrafo, sotto-sottoparagrafo, capoverso e sottocapoverso; in inglese esse si chiamano rispettivamente *section, subsection, subsubsection, paragraph, subparagraph.* Si noti la presenza dei falsi amici 'paragrafo' e 'paragraph'!

Il capoverso è quel tratto di testo che comincia (generalmente con una rientranza) con una lettera maiuscola e termina con un 'punto e a capo'. Un capoverso può contenere diversi periodi, ciascuno diviso in diverse frasi, a loro volta... ma qui si sconfinerebbe nella grammatica della lingua. Spesso singoli periodi, o anche solo delle frasi di una certa lunghezza, sono numerati e, come sotto-capoversi, a seconda dello scritto possono chiamarsi versetti, oppure commi. In generale i versetti sono solo numerati, mentre i commi possono avere anche un brevissimo titolo; in questo caso essi cominciano su una nuova riga di testo, ma nei testi di carattere legale il "capoverso" (che allora si chiamerà 'articolo') viene distinto dal fatto che viene inserito un visibile contrografismo e un titolo, come minimo un numero, fra un articolo e il successivo.

Finito il corpo del testo, cioè dopo l'ultimo capitolo, comincia il materiale finale, la *back matter*, la cui numerazione araba prosegue quella del corpo del testo, ma la numerazione dei capitoli, spesso chiamati 'appendici', ricomincia da 1, o meglio da A, visto che si usa una numerazione diversa, per esempio letterale; ma mentre i capitoli e le parti cominciano sempre sul recto delle pagine, nel materiale finale i capitoli possono cominciare anche sul verso.

Il materiale finale contiene la Bibliografia, gli eventuali glossari, le appendici (capitoli numerati con lettere), e termina, se ci sono, con uno o più indici analitici, i quali vengono evidentemente composti per ultimi per evitare che le loro informazioni siano errate a causa delle correzioni eventualmente apportate nel corpo del testo.

L'ultimissima pagina, sul verso, potrebbe contenere il colophon, una pagina, cioè, nella quale sono descritte le particolarità compositive del testo, i materiali usati e simili altre informazioni di carattere tipografico. Vengono poi lasciate un paio di pagine bianche per poter inserire i risguardi posteriori a cui è incollata la parte posteriore della copertina.

[3] LaTeX di default usa i numeri romani minuscoli scrivendo per esempio l'equivalente di 1666 come 'mdclxvi'; in Europa, tranne nelle isole britanniche, generalmente si preferisce il maiuscoletto; questa modifica alla classe `book` è una delle pochissime che sono state eseguite per la composizione di questo testo.

Per ovvi motivi economici le varie pagine di un libro sono ottenute piegando un certo numero di volte dei grandi fogli; queste *segnature* contengono evidentemente un numero di facciate pari ad una potenza di 2; ci saranno perciò libri in ottavi, in sedicesimi, in trentaduesimi, raramente con un numero di facciate superiore, perché lo spessore delle segnature produrrebbe problemi.[4] Resta il fatto che, tolti i risguardi, le pagine interne devono essere un multiplo del numero di pagine di una segnatura; al massimo l'ultima segnatura di un libro formato da segnature di 32 pagine, potrà essere una segnatura da 16 pagine o anche di 8 pagine, ma questo aumenta leggermente i costi di produzione. Nel predisporre la produzione di un libro le tipografie generalmente sono più attente al numero di pagine che non al loro contenuto (il contenuto è responsabilità dell'autore e della casa editrice); ma è bene che anche l'autore conosca questi problemi in modo da sapersi regolare se deve produrre un testo pronto per la stampa, cioè un file PDF da trasmettere alla tipografia che non vi mette mano se non per ottenere dal file le lastre per la stampa delle due facce dei grandi fogli che formeranno ciascuna segnatura.

2.7 Osservazioni finali

È chiaro che queste poche pagine non sono in grado di dare altro che una breve panoramica della terminologia tipografica, ma spero che il lettore, con queste poche conoscenze, sia stimolato ad osservare con più attenzione i libri che ha per mano e cerchi di valutare i vari elementi descritti, sia in relazione ai caratteri, sia in relazione agli spazi e alla suddivisione dello stampato.

I nomi inglesi che ho indicato servono anche per conoscere il significato di molti comandi o istruzioni del linguaggio LaTeX, che usa quelle parole o quelle brevi locuzioni, ridotte ad una sola stringa senza spazi, per indicare esattamente quegli oggetti individuati dai nomi inglesi.

[4] Con particolari piegature si potrebbero avere anche segnature il cui numero di facciate corrisponde ad una potenza di due moltiplicata per tre; per esempio si potrebbero avere segnature di 6 pagine, di 12 pagine, eccetera; questi tipi di piegature non sono molto frequenti nei libri, ma si trovano usate in fascicoletti propagandistici e altri simili opuscoli.

Capitolo 3

Ortografia tipografica

Che cosa si intende per ortografia tipografica? Tutti conoscono il significato della parola *ortografia*: la scrittura corretta. Ma questo significato di solito viene inteso in senso limitativo come la scrittura corretta delle singole parole.

Quando si scrive con un potente word processor o con un programma di composizione come LaTeX le regole da osservare sono molto più numerose perché molto più numerosi sono i modi di scrivere sia per l'abbondanza dei segni a disposizione, sia per le famiglie, serie, forme, corpi dei caratteri, sia per la posizione dei segni, sia per la disposizione del testo sulle pagine, sia per l'interpunzione; insomma, la tipografia è un'altra cosa.

È ovvio che l'ortografia tipografica, quindi, comprende anche l'ortografia testuale, ma la scrittura tipografica corretta richiede di rispettare anche altre regole.

Se ne è accennato anche in altri capitoli; alcuni degli argomenti verranno, quindi, ripresi e sviluppati con maggiore respiro, cercando di spiegare il perché di certe regole, anche per imparare a deviare coscientemente da quelle esposte o per dirimere la questione delle ambiguità: "in questo caso si può fare così, oppure così"; va bene, ma allora che cosa è meglio fare? Conoscendo i perché si può trovare la risposta a ragion veduta.

3.1 Ortografia testuale

Non c'è bisogno di raccomandare la scrittura corretta di ogni parola del testo; è del tutto ovvio. Ma la pratica insegna che durante l'introduzione del testo si compiono numerosi errori di battitura, anche se il "tastierista" conosce benissimo l'ortografia. Si suole dare il nome di *refusi* a questo tipo di errori, perché ai tempi della composizione con i caratteri metallici spesso era necessario fondere di nuovo i caratteri di una o più righe al fine di eseguire la correzione.

Oggi ogni programma di elaborazione di testo permette di fare la verifica ortografica; spesso la verifica può essere fatta solo su testo evidenziato in modo

da poter usare regole ortografiche diverse per lingue diverse. Tuttavia è ben noto che questi correttori ortografici permettono di eliminare un gran numero di refusi, ma non permettono di eliminare quei refusi che generano altre parole valide; i famosi *fischi* e *fiaschi*; *corpo* e *copro*; *viene* e *vine*, anche insolite[1]; l'elenco potrebbe andare avanti a piacere. Ne consegue che la rilettura del testo va fatta da una persona possibilmente diversa dal "tastierista", magari un correttore di bozze professionista; in ogni caso la rilettura non va fatta subito, ma bisogna lasciar trascorrere alcuni giorni e anche il "tastierista" può rendersi conto di altri refusi sfuggiti ai precedenti controlli.

3.1.1 Ortografie alternative

Esistono molte parole che ammettono diverse ortografie, per esempio: *obbiettivo* e *obiettivo*; *uguale* e *eguale*; in questi casi è necessario verificare su un buon dizionario della lingua italiana per verificare se per caso non esistano delle preferenze per l'una o l'altra scrittura, se non esistano degli ambiti testuali nei quali sia preferita una scrittura e altri ambiti nei quali sia preferita l'altra.

Ortograficamente in italiano sono soggetti ad oscillazioni i plurali maschili delle parole che terminano in *-co*, in *-go* e in *-io* e delle parole femminili che terminano in *-cia* e *-gia*. Esistono delle regole sia per i nomi maschili sia per quelli femminili, ma queste regole presentano numerose eccezioni; quindi in caso di dubbio è meglio riferirsi a un buon dizionario.

Per i maschili in *-co* e *-go* vale la regola di default che le parole piane mantengono il suono gutturale (e quindi viene inserita una 'h'), mentre le parole sdrucciole perdono la pronuncia gutturale e quindi non prendono nessuna 'h': *chirurgo* produce *chirurghi*, *fico* produce *fichi*, ma *amico* produce *amici*. Invece *psicologo* produce *psicologi*, *elettrico* produce *elettrici*, eccetera.

Per i maschili terminanti in *-io* con la 'i' tonica (quindi *io* non forma un dittongo) non ci sono problemi e la 'o' muta in 'i' e si hanno quindi due 'i' consecutive come terminazione: *zio* produce *zii*; *formicolio* produce *formicolii*.

Ma per i maschili terminanti con il dittongo ascendente *-io*, una volta i plurali venivano eseguiti con due 'i' (*studii*), oppure con una 'i' segnata da un circonflesso (*studî*), oppure con la desinenza *j* (*studj*) e persino *-ij* (*studij*); oggi, si preferisce generalmente scrivere il plurale con una sola 'i', assumendo che la 'vocale' terminale 'i' assorba la 'semivocale' precedente 'i'. Tuttavia si mantengono le due 'i' anche modernamente quando certi plurali si potrebbero confondere: *assassinio* produce *assassinii* perché questa parola non sia confusa con il plurale di *assassino*; nonostante lo spostamento dell'accento anche *desiderio* produce *desiderii* per non confondere con la seconda persona del presente del verbo 'desiderare', *desideri*; se si usasse sistematicamente l'accento sulle parole sdrucciole non ci sarebbe nessun motivo per derogare dalla regola 'normale'. Di nuovo, nel dubbio è meglio consultare un buon dizionario.

[1] La parola *vine* è il plurale di *vina*, nome di uno strumento musicale indiano...

Per i femminili in *-cia* e *-gia*, purché la 'i' non sia tonica, i plurali oggi si compongono quasi senza eccezioni conservando la 'i' semiconsonantica se la sillaba finale è preceduta da una sillaba aperta, mentre al contrario la perdono se essa è preceduta da una sillaba chiusa[2]: *camicia* produce *camicie*, ma *bilancia* produce *bilance*; *valigia* produce *valigie*, ma *cengia* produce *cenge*. A questa regola della sillaba precedente aperta o chiusa sembrano[3] sfuggire le parole terminanti in *-scia* che hanno costantemente il plurale in *-sce*: *ascia* e *asce*; *biscia* e *bisce*; *striscia* e *strisce*. Come sempre la consultazione di un buon dizionario permette di risolvere eventuali eccezioni a questa regola; infatti *ciliegia* ammette entrambi i plurali *ciliegie* e *ciliege*. Il dizionario della lingua italiana di Aldo Gabrielli (Hoepli) indica entrambi i plurali, ma negli esempi usa solo la desinenza *-gie*.

Anche scrivendo in lingue straniere avvengono gli stessi fenomeni, con l'aggravante che esistono circostanze nelle quali in una nazione si usa una certa ortografia diversa da quella di un'altra nazione. Tipica è la differenza fra l'inglese americano e l'inglese britannico, dove non solo alcune parole omografe hanno significati completamente diversi, ma parole molto comuni hanno ortografie diverse, per esempio: *colour* e *color*; *behaviour* e *behavior*; *to analize* e *to analise*.

In queste circostanze il compositore deve scegliere una delle grafie alternative e usare sempre quella; per il lettore non c'è nulla di più fastidioso del trovare la stessa parola scritta in modi diversi. Ovviamente, se si cita un brano altrui, deve essere rispettata anche la sua ortografia, ma il fatto di avere evidenziato la citazione fa capire al lettore che quelle parole sono scritte da un'altra persona.

3.1.2 'd' eufonica e 'i' prostetica, apostrofi

Nel parlare si usano spesso le 'd' eufoniche e le 'i' prostetiche. Nel parlare si pronuncia spesso una 'd' fra una congiunzione o una preposizione terminante in vocale e la parola seguente specialmente quando inizia con la stessa vocale. Tipiche le pronunce di *ad*, *ed* e *od*. Nello scrivere è generalmente accettata la 'd' eufonica per evitare lo 'scontro' di due 'a' o di due 'e': *ad altro scopo*; *ad avere*; *ed essere*. È anche accettata la scrittura di *ad esempio*, negli altri casi la 'd' eufonica dovrebbe essere evitata.

Questa norma vale a maggior ragione per la 'i' prostetica, quella 'i' che viene spesso pronunciata dopo preposizioni che finiscono con una consonante se precedono parole che cominciano con la 's impura': *in iscuola*; *in istrada*. Anche se questa 'i' viene pronunciata, non dovrebbe venire mai scritta; si tollera la locuzione *per iscritto*.

[2]Una sillaba è aperta se termina con una vocale, mentre è chiusa se termina con una consonante.

[3]Sembrano soltanto se si prende la definizione in senso troppo letterale: è vero che *cia* è preceduto da *s*, ma il nesso *sci* si riferisce ad una pronuncia particolare diversa dalla pronuncia di *s* seguito dalla pronuncia di *cia*; in alcuni dialetti italiani o pronunce dialettali, non è infrequente questa pronuncia separata e l'ortografia italiana non è predisposta per renderla in modo semplice: in piemontese si trova, per esempio, *s-centrà*, ma in molte parti di Italia si sente pronunciare in 'italiano' *s-centrato*.

Probabilmente esistono simili variazioni di grafia anche in altre lingue; è quindi molto opportuno conoscere a fondo la lingua in cui si scrive, oppure, citando brani altrui, bisogna essere certi di riprodurli integralmente nella lingua in cui sono scritti (eventualmente una traduzione in nota potrebbe essere molto gradita al lettore).

L'apostrofo viene impiegato abbondantemente in italiano, in francese, in catalano, anche nella variante valenzana, e in retoromanzo per indicare l'elisione di una vocale; in italiano bisogna stare attenti a non confondere l'elisione con il troncamento; sarebbe sbagliatissimo scrivere *un'uomo* dove invece l'articolo indeterminativo maschile subisce solo il troncamento (la semplice caduta della 'o') e di fatto è un errore rarissimo negli scritti a stampa. Invece è piuttosto frequente incontrare espressioni errate come *qual'è*.

Salvo poche locuzioni ormai cristallizzate, l'elisione si manifesta in italiano solo con gli articoli determinativi maschili e femminili e con gli articoli indeterminativi femminili. Ma oggigiorno l'elisione viene eseguita raramente, limitandola agli articoli singolari e anche in questi casi spesso non viene eseguita: si trova quasi sempre *l'equazione* o talvolta, nei testi tecnici, *la equazione*, ma non si incontra mai *l'equazioni*. Sessanta anni fa era comune vedere scritto *gl'italiani*; oggi non si incontra più questa scrittura. Vista la possibilità di eseguire o di non eseguire l'elisione, il compositore deve decidere come comportarsi e poi deve coerentemente seguire la sua decisione; come al solito il lettore apprezza la coerenza.

L'apostrofo viene anche usato per marcare apocopi e aferesi che oggi in italiano si incontrano poco: *un po' scuro*; *'sta volta*. In espressioni da recitare si usa l'apostrofo per indicare gli imperativi di alcuni verbi: *da' qui*; *fa' attenzione*[4]. In passato era molto comune scrivere *e'* per indicare il pronome maschile di terza persona *ei*, ma quest'uso è durato solo nei primi secoli di vita dell'italiano. Oggi è corretto scriverlo solo nelle citazioni di brani tratti da testi antichi.

In tipografia non bisogna confondere l'apostrofo con l'apice; molti font dispongono di due segni distinti; con i font normalmente usati con LaTeX si usa un solo segno nel testo (’), mentre in matematica il segno (′) è diverso, ma ci pensa sempre LaTeX ad usare i segno giusto se nel file sorgente si usa sempre il segno dell'apostrofo.

3.2 Accenti

In italiano gli accenti obbligatori sono da porre solo sull'ultima vocale delle parole *tronche*; una apposita norma UNI regola la questione, ma in buona sostanza tutte le vocali terminali delle parole tronche ricevono l'accento grave, tranne la 'e' che può riceverlo o grave o acuto secondo precise regole ortografiche.

L'accento acuto (´) si pone su *ché* e sui suoi derivati *perché*, *affinché*, eccetera; sui derivati di *tre*, *re* e *fe'* (fede): *ventitré*, *viceré*, *autodafé*, eccetera; su alcune parole come *scimpanzé*, *sé*.

[4] Si noti la differenza: *dà*, terza persona singolare dell'indicativo di *dare*; *da'*, seconda persona singolare dell'imperativo di *dare*.

L'accento grave (`) si pone sulla terza persona singolare del verbo essere *è* e sui suoi derivati come *cioè*; sulle parole acquisite in italiano e derivate prevalentemente dal francese, come *bidè*, *relè*; su nomi di bevande come *caffè*, *tè* (la scrittura *the* è errata); sui derivati di *me* come *ahimè*; sui nomi propri come *Noè*, *Mosè*, *Giosuè*, *Josè* (anche se in spagnolo si scrive *José*, che però è ugualmente corretto), eccetera.

Si fa osservare che gli italiani sono piuttosto sciatti nel distinguere gli apostrofi grave e acuto; non è raro trovare scritto *perchè*, comprensibile per il fatto che con la tastiera italiana la 'è' si ottiene premendo un solo tasto, ma si trova scritto anche *é*, che non è nemmeno comprensibile perché con la tastiera italiana bisogna premere due tasti. Il compositore e il "tastierista" che vuol rispettare l'ortografia tipografica, oltre che quella grammaticale, deve quindi porre particolare attenzione alla distinzione di questi due accenti.

La norma UNI è piuttosto poco impegnativa, nel senso che dà indicazioni poco restrittive; essa consente per esempio di usare l'accento acuto sulle vocali 'i' e 'u', vocali naturalmente chiuse, per le quali sembrerebbe più adeguato l'uso dell'accento acuto, che specifica il suono chiuso delle vocali 'e' e 'o'. Mentre sulla 'e' la norma è chiara (e nel caso di dubbio è consigliabile rivolgersi ad un buon dizionario), sulla 'o' delle parole tronche il suono è sempre aperto, quindi l'accento obbligatorio sulla 'o' è sempre grave.

Il filologo italiano Gian Luigi Beccaria usa regolarmente gli accenti acuti sulla 'i' e sulla 'u' (*cosí, piú*), cosa che è permessa anche dalla norma UNI, ma con la tastiera italiana è difficile da realizzare; egli usa anche il circonflesso sui plurali dei maschili terminati in *-io* (*principî*).

Ma la norma UNI consente anche gli accenti facoltativi, cioè sulle sillabe interne. In questo caso, se si vuole distinguere *cólto* (istruito) da *còlto* (raccolto) bisogna scegliere l'accento secondo l'apertura della vocale 'o'; è quanto mai opportuno verificare la pronuncia corretta su un buon dizionario moderno, dove, spesso, la pronuncia è descritta mediante i segni dell'alfabeto IPA (International Phonetic Association).

Il filòlogo Migliorini, membro della commissione che curò la redazione della norma UNI in questione, vi fece introdurre la specificazione che non si accéntano mai le parole piane, se non per indicare il suono aperto o chiuso della vocale tònica 'e' oppure 'o'; si pòssono accentare, e si consiglia di farlo, le parole sdrùcciole e bisdrùcciole[5]; il consiglio diventa quasi un imperativo quando vi possa essere confusione con parole omògrafe con pronuncia piana. Seguendo questa indicazione si distinguerà *prìncipi* da *principi*, ma non si scriverà mai *princìpi*; si distinguerà *séguito* da *seguito*, ma non si scriverà mai *seguìto*; a maggior ragione si distinguerà *làvati* (imperativo) da *lavati* (participio) e a nessuno verrebbe in mente di scrìvere *lavàti*. Tuttavia si incóntrano spesso parole piane accentate in violazione dell'indicazione espressa nella norma UNI. Anche il rispetto di questa norma costituisce un aspetto dell'ortografia tipogràfica.

[5] È quello che si è fatto in questo capoverso.

Non era così un tempo. Su un libro stampato da Bodoni nel 1803 ho trovato che l'accentazione dell'italiano era eseguita con le regole del greco: accento grave sulle parole tronche e accento acuto sulle parole sdrucciole o bisdrucciole, indipendentemente dall'apertura della vocale. Ovviamente una citazione di quel testo richiede il rispetto di quel tipo di accentazione; ma il fatto che il testo sia messo in evidenza come citazione, giustifica il fatto di non essere conforme alla norma UNI (che non esisteva nel 1803!).

3.3 Sillabazione

Secondo alcuni sarebbe meglio comporre in modo che nessuna parola sia divisa in sillabe in fin di riga. Probabilmente è vero, ma la necessità di eseguire la cesura dipende dalla lunghezza della riga di testo e dalla lunghezza della parola; sicuramente non fa parte della 'ortografia' tipografica avere diverse righe *consecutive* che terminano con la lineetta di 'a capo'. Se la giustezza è decisamente piccola è meglio comporre in bandiera allineata a sinistra, piuttosto che con la giustificazione da entrambi i lati. È qui che il compositore deve esercitare il suo giudizio per comporre 'giusto'.

Tuttavia una delle cose che salta all'occhio quando si legge un testo con molte cesure, è non solo se queste cesure siano corrette (ci mancherebbe altro!) ma se sono eseguite in modo da rendere confortevole la lettura. È per questo, per esempio, che le regole di sillabazione realizzate da LaTeX per la lingua italiana evitano di dividere i possibili dittonghi, i gruppi di vocali che potrebbero essere tali, ma che, non sapendo leggere e non sapendo dove cade l'accento, LaTeX non può trattare con certezza. In questo modo si perdono possibili punti di cesura, ma la lettura resta molto più confortevole. Si pensi ad una parola come *paura*; le lettere *au* in altre parole formano un dittongo discendente, ma qui formano uno iato e la parola, grammaticalmente, si potrebbe dividere in *pa-u-ra*. LaTeX divide invece solamente in *pau-ra*. In effetti non sarebbe sbagliato se LaTeX dividesse come dice la grammatica, ma il lettore si troverebbe a disagio se la riga finisse con 'pa-' e la riga successiva iniziasse con 'ura'. Questo genere di problemi avviene più spesso di quanto non si immagini, ma la sillabazione tipografica (e quindi l'ortografia tipografica) differisce dalla sillabazione ordinaria (e quindi dall'ortografia ordinaria) proprio perché tende ad avere anche il lettore, non solo la grammatica, come metro di misura della 'correttezza'.

Bisogna ricordarsi di selezionare la lingua giusta con `babel` in modo che LaTeX possa usare le regole di sillabazione adatte alla lingua in uso; bisogna ricordarsi di specificare la lingua ogni volta che si compone in una lingua diversa da quella principale; forse il testo in lingua non subirà nessuna cesura in fin di riga, ma non lo si può sapere in anticipo, quindi è meglio provvedere per tempo.

Se componendo in colonne strette, si vuole aggiungere qualche punto di divisione in fase di correzione delle bozze, si può sempre inserire nell'argomento del comando `\hyphenation` una lista di parole con iati divise tenendone conto;

per esempio si potrebbero inserire (con tutte le varianti e sempre che le parole siano effettivamente usate nel testo):

```
\hyphenation{pa-u-ra pa-u-re i-de-a-le i-de-a-li ba-u-le ba-u-li
             cro-a-to cro-a-ti cro-a-ta cro-a-te}
```

Anche le parole scientifiche ottenute dall'agglutinazione di varie radici è bene che siano divise etimologicamente per facilitarne la comprensione da parte del lettore; le norme UNI che regolano la questione sono piuttosto flessibili in questo senso; esse danno le regole generali per la divisione fonetica, ma non proibiscono al compositore di eseguire la divisione etimologica. Se una parola scientifica di questo genere capita sovente nel testo, è meglio inserirla con le sue variazioni nella lista di `\hyphenation`; se la si incontra sporadicamente può essere più semplice introdurre una divisione opzionale mediante l'uso del carattere attivo `"`, che consente la divisione anche negli altri 'monconi' della parola, piuttosto che il comando LaTeX `\-`, che invece consente la divisione solo nel punto dove è indicato. Il carattere attivo `"` produce questo effetto solo mentre si compone in italiano[6]; non è possibile usarlo componendo in altre lingue, per le quali esistono comandi analoghi formati spesso da due segni.

3.4 Punteggiatura

Il problema della punteggiatura si presenta sia nella composizione del testo sia quando si inseriscono parti matematiche; se ne parla diffusamente anche in altri capitoli. Si tenga presente che i segni di punteggiatura non sono solo i quattro o sei segni ai quali siamo soliti dare questi nomi. In tipografia ci sono:

1. i segni di marcatura logica: la virgola, il punto e virgola, i due punti e il punto.

2. I segni di espressione: il punto interrogativo, il punto esclamativo e i puntini di sospensione (usati anche per le omissioni).

3. I segni di enfasi: le virgolette alte, le virgolette basse (in entrambi i casi semplici o doppie).

4. I delimitatori: principalmente le parentesi tonde e quadre, ma anche le coppie di virgole.

5. I segni di sezionamento o di opposizione: la barra obliqua, il trattino o lineetta, il trattino medio e il trattino o lineato lungo.

[6] Quando si usa xelatex e si specifica *babel* come pacchetto da usare per la gestione delle lingue, il doppio apice dovrebbe poter essere usato come descritto qui; se invece si gestiscono le lingue con *polyglossia* il doppio apice non è più attivo e non può venire usato per la divisione in sillabe su base etimologica, a meno che specificando la lingua italiana con i comandi proprio di *polyglossia*, non si specifichi l'opzione *babelshorthands*.

6. I segni di sezionamento usati anche come simboli a sé stanti: il segno di paragrafo, §, e il segno di capoverso, ¶.

Direi che quelli più difficili da usare correttamente sono i semplici segni di marcatura logica. Infatti fin dalle scuole elementari siamo stati tutti esposti alla definizione di tali segni come dei segni che marcano le pause. Non è vero; la virgola, il punto, i due punti e il punto e virgola servono per marcare le divisioni sintattico-grammaticali del testo secondo regole che nulla hanno a che vedere con la marcatura prosodica. Questa infatti non può essere marcata adeguatamente con i soli quattro segni indicati, e ogni oratore che debba leggere un discorso ha la sua personale maniera di indicare i segni prosodici per modulare le inflessioni della voce, per marcare le pause d'effetto, eccetera. Questo naturalmente non vuol dire che i segni di interpunzione non contribuiscano alle pause, se uno dovesse leggere ad alta voce, ma non generano necessariamente pause brevi, medie o lunghe in corrispondenza alla virgola, al punto e virgola, e al punto; possono invece aiutare a mutare il tono della voce. Tuttavia queste pause e questi mutamenti non sono la funzione principale della punteggiatura, sono solo l'aiuto alla comprensione per il lettore che esegue le pause e i mutamenti in base a quello che egli ha compreso. Dare la definizione dei segni di punteggiatura mediante la durata delle pause significa scambiare la causa con l'effetto.

I segni prosodici servono per leggere ad alta voce un testo. I segni di punteggiatura servono per dare risalto alle sezioni sintattico-grammaticali del testo e servono, in sostanza, ad aiutare il lettore a comprendere lo scritto leggendo con gli occhi, non con la bocca; gli occhi non hanno bisogno di respirare.

La virgola serve in generale some separatore seriale nelle enumerazioni, serve per separare certe frasi subordinate dalla frase principale; usata a coppie, serve per marcare gli incisi, può essere usata per spostare l'attenzione del lettore da una parola ad un altra anche a poca distanza. Le coppie di virgole delimitatrici di un inciso possono non essere apparenti, perché se risultano adiacenti ad un segno di interpunzione di maggiore importanza, ne vengono assorbite.

La funzione seriale è evidente: "Chiare, dolci, fresche acque...".

La funzione di separazione di frasi non è altrettanto semplice da illustrare; ci sono frasi subordinate che si pensano sempre come subordinate e quindi separate da una virgola, ed altre che non svolgono sempre lo stesso ruolo, anche se la grammatica le classifica nello stesso modo.

Per esempio l'ipotesi di una costruzione ipotetica è quasi sempre separata con una virgola dalla tesi: "Se arrivo in tempo, posso cenare con voi." Invece una frase relativa viene separata con una virgola dalla principale solo se ha carattere esplicativo o appositivo, mentre non è separata dalla virgola se ha significato specificativo o restrittivo: "Il paziente che ha manifestato una forte poliuria va trattato con..."; la frase relativa è specificativa nel senso che la sua eliminazione toglierebbe significato a quello che resta. Viceversa una relativa è esplicativa quando la sua eliminazione toglie qualcosa dal discorso ma la sua assenza non preclude la comprensione del testo restante. La professoressa Mortara Garavelli in [42] mostra un bell'esempio di relative che svolgono ruoli diversi: le parole sono le stesse, mentre i periodi differiscono solo per una virgola:

- Non seguo i programmi televisivi che mi sembrano scadenti.
- Non seguo i programmi televisivi, che mi sembrano scadenti.

I due ruoli della frase relativa sono assolutamente evidenti; nel primo caso il soggetto non segue alcuni programmi televisivi, quelli che gli sembrano scadenti; nel secondo caso il soggetto non segue nessun programma televisivo e spiega che non li segue perché tutti i programmi gli sembrano scadenti.

La virgola serve anche per spostare l'attenzione. Poiché nella frase ordinaria italiana i concetti più "pesanti" vengono collocati alla fine, introducendo una interruzione con una virgola, è possibile spostare l'attenzione su un concetto precedente; si osservi ancora la coppia di frasi proposte dalla professoressa Mortara Garavelli:

- poveri ma belli
- poveri, ma belli

Nel primo caso l'attenzione cade su "belli", mentre nel secondo caso cade su "poveri".

Ecco quindi che molte regole delle grammatiche elementari diventano delle costrizioni che impediscono di comunicare correttamente il pensiero in forma scritta.

Forse anche il problema della punteggiatura esterna alla matematica in display potrebbe essere visto alla luce di queste considerazioni. Può darsi che lo schieramento a favore di uno o l'altro dei due approcci comprenda che l'altro approccio è più conforme alla punteggiatura in generale; può darsi, anzi, che entrambi gli schieramenti si orientino verso una interpunzione esterna alla matematica che non sia né la totale assenza di interpunzione, né la presenza di una interpunzione abbondante. Qui non si prende posizione nel modo più assoluto; si vuole semplicemente osservare che l'interpunzione va vista con uno sguardo più ampio di quello che le grammatiche scolastiche hanno postulato, e che noi tutti ci portiamo dietro fin dai tempi delle scuole elementari.

Il punto e virgola svolge o può svolgere funzioni simili a quelli della virgola, ma solitamente serve per separare frasi autonome grammaticalmente; nella sua funzione di serializzazione serve per separare frasi che contengano esplicitamente o implicitamente anche il verbo. Il ruolo di enfasi è ancora più importante con il punto e virgola che con la virgola.

Il punto serve per terminare un periodo; se dopo si va a capo esso segna anche la fine di un capoverso; può servire anche per le abbreviazioni.

I due punti sono un po' particolari: essi servono per introdurre una frase che sia una spiegazione della precedente; comunque costituisce un modo per creare una aspettativa per ciò che segue.

Vale la pena di ricordare che in inglese la virgola, i due punti e il punto si chiamano rispettivamente *comma*, *colon* e *period*. Queste sono le forme rimaste nella lingua inglese delle denominazioni latine che venivano usate nei manuali di oratoria, come quello di Quintiliano. In latino vigeva la scrittura continua,

senza nemmeno gli spazi fra le parole. Solo verso la metà del primo millennio si cominciò a scrivere, almeno nelle iscrizioni sulle lapidi, inserendo un punto a mezza riga fra una parola e l'altra. Per i non latinisti è piuttosto difficile leggere iscrizioni latine con la scrittura continua. Lo doveva essere anche per gli antichi, e Quintiliano insegnava a distinguere le locuzioni dalle frasi e dai periodi (rispettivamente in latino: *comma*, *colon* e *periodus*) mediante dei segni di interpunzione; il comma veniva terminato con un punto basso, il colon con un punto medio e il periodo con un punto alto. Questi erano allo stesso tempo segni demarcatori sintattici e segni prosodici, ma Quintiliano insegnava anche che, per enunciare correttamente il discorso, bisognava prima capirlo, poi lo si poteva leggere ad alta voce.

Nei secoli questi tre segni hanno cambiato forma e funzione; nel testo [43] sono descritte queste evoluzioni in tutte le lingue europee principali; si tratta di una lettura molto interessante, anche se può sembrare una lettura piuttosto pesante. I collaboratori della curatrice in varia forma hanno esposto i loro studi, alcuni in un registro da iniziati, altri in un registro più divulgativo, ma tutti in modo interessante.

In italiano la virgola (da *virgula*, diminutivo latino di *verga*, asta), svolge, tra le altre, la funzione del comma. Gli altri segni hanno nomi descrittivi della forma del segno, ma il punto e virgola svolge il ruolo del colon, mentre i due punti sono particolari nel compito di introdurre una nuova frase. In italiano la parola *comma* è rimasta solo ad indicare una sola modesta articolazione del testo, spesso un solo periodo, di un articolo di legge. Oggi il termine *colon* in italiano indica un segmento ritmico di un testo, individuato da pause metriche o logiche (non necessariamente marcate con segni di interpunzione).

Va ancora notato che ci sono alcune cose fortemente vietate nella punteggiatura, come per esempio separare il soggetto dal verbo con una virgola; apparentemente il Manzoni lo faceva, ma ad una seconda lettura si capisce che le sue virgole hanno lo scopo di marcare certe enfasi che solo apparentemente separano il soggetto dal verbo; ecco un criticatissimo esempio tratto dai *Promessi Sposi*:

> Però, di tante belle parole Renzo, non ne credette una: né che il notaio volesse più bene a lui che a' birri, né che prendesse tanto a cuore la sua reputazione [...].

Oggi noi (che non siamo Manzoni) considereremmo questa frase errata e metteremmo la virgola prima di 'Renzo'. In realtà, leggendo meglio, sembra di guardare in faccia il narratore che, arrivato a questo punto del racconto, strizza l'occhio come a dire: "mica stupido il ragazzo!".

In inglese americano è invece comune incontrare questa funzione prosodica di separazione del soggetto dal verbo, ma non sembra un esempio da imitare nemmeno in inglese.

Qui non è il caso di riscrivere il Prontuario della professoressa Mortara Garavelli [42], a cui reindirizziamo il lettore. Però si sottolinea l'importanza della

punteggiatura nella buona tipografia; interpungere bene componendo tipograficamente può voler dire trasmettere un messaggio piuttosto che un altro, evitando di travisare il senso del discorso rispetto a come l'aveva in mente l'autore.

Per i segni di espressione, il punto interrogativo '?' ed esclamativo '!', talvolta geminati in '?!', non dovrebbero esserci problemi interpretativi; per i puntini di sospensione è chiara la loro funzione; tipograficamente parlando in Italia si suole scrivere i puntini di sospensione attaccati alla parola precedente; nella tipografia anglo-americana i puntini si scrivono attaccati se rappresentano una sospensione, mentre si scrivono staccati se rappresentano una omissione. Nella tradizione tipografica italiana i tre puntini restano tre anche alla fine del periodo, nel senso che conglobano anche la funzione di punto fermo.

I demarcatori come le virgolette alte o basse non richiedono spiegazioni particolari; tuttavia, disponendo di entrambi i tipi di segni, si è soliti dedicare le virgolette basse (chiamate anche: caporali, sergenti, virgolette uncinate, virgolette quadrate) per evidenziare parole o locuzioni importanti, ovvero per racchiudere i dialoghi; le virgolette alte (o inglesi) allora si usano per evidenziare parole usate come tali oppure per segnalarne un uso ironico o traslato, comunque inconsueto. Le brevi citazioni nel corpo del testo vanno racchiuse fra virgolette; una breve citazione che ne contenga un'altra richiede l'uso di virgolette diverse: la citazione interna avrà per esempio le virgolette semplici e quella esterna le virgolette doppie; oppure quella esterna le virgolette basse e quella interna le virgolette alte.

I problemi si complicano dal punto di vista della correttezza tipografica quando si devono evidenziare i dialoghi, per esempio, in un romanzo. Ci sono diversi modi di evidenziare i dialoghi e le battute non facenti parte propriamente di un dialogo. Possono essere introdotte da un trattino medio, possono essere racchiuse fra virgolette basse, come detto sopra, possono racchiudere un'altra battuta; possono essere solo pensate ma non dette, eccetera. Le cose si possono complicare molto. Si veda più avanti anche in merito all'uso del trattino.

Come al solito la coerenza è il metro principale per eseguire le scelte tipografiche giuste. I trattini, le virgolette alte o basse, semplici o doppie sono 'codici' per distinguere le battute.

Un modo coerente per usare questi segni è il seguente:

- si usano le virgolette basse doppie aperte e chiuse per racchiudere ciascuna battuta detta;
- si usano i trattini per separare incisi narrativi all'interno di una battuta detta, per esempio: «Sì, è proprio così, – affermò con decisione – ne sono assolutamente sicura.»
- si usano le virgolette alte doppie aperte e chiuse per racchiudere una battuta dentro un'altra battuta, per esempio: «Mi hai detto testualmente "Cattivo!" con un tono troppo aggressivo.»
- si usano le virgolette alte doppie aperte e chiuse per racchiudere una battuta pensata, per esempio: . . . pensai: "Ora qui devo cavarmela da solo."

- si usano le virgolette alte semplici aperte e chiuse per racchiudere una battuta dentro una battuta pensata, per esempio: . . . pensai: “Gianni mi ha detto ‘Sì’ con poca convinzione.”

Chiaramente questo schema è coerente, ma non è l’unico che si possa usare.

A differenza del francese e di altre lingue, oltre che dalla pratica tipografica italiana dei secoli scorsi, le virgolette alte o basse non richiedono mai spazi aggiuntivi al loro interno; quelle di apertura sono precedute da uno spazio, mentre quelle di chiusura sono seguite da uno spazio o da un segno di interpunzione, seguito a sua volta dallo spazio necessario.

Si faccia attenzione nelle citazioni in lingua straniera, dove le virgolette vengono usate come nella lingua d’origine; in alcune nazioni le virgolette sono usate con la concavità verso l’esterno; in alcune nazioni le virgolette di apertura sono della stessa forma delle virgolette alte di chiusura ma sono poste alla base della riga di stampa; in alcune nazioni le virgolette vengono ripetute all’inizio di ogni riga per le citazioni che durino più righe. Come già detto, in francese le virgolette sono distanziate con normali spazi interparola dal loro contenuto, mentre in italiano questo oggi è assolutamente vietato. Nel XIX secolo anche in Italia andava per la maggiore lo stile tipografico francese, per cui anche in Italia le virgolette erano distanziate, così come anche i segni di interpunzione, compresi la virgola e il punto. Oggi sarebbe inaccettabile. Anche in latino ecclesiastico[7] si usa uno stile tipografico del tipo francese, ma con spazi più ridotti rispetto a quelli francesi.

Le parentesi tonde servono per incisi più importanti di quelli che si possono delimitare con una coppia di virgole; anche le parentesi tonde non vogliono spazi al loro interno. Le parentesi quadre servono per inserire delle correzioni, specialmente nelle citazioni; delimitano, insomma, delle parti che nella citazione originale mancavano o erano diverse. Le parentesi tonde e quelle quadre possono avere anche altre funzioni, come quelle di delimitare le numerazioni delle formule o delle citazioni bibliografiche; sarebbe importante per una buona e corretta tipografia che le parentesi di ogni tipo non svolgessero più di due ruoli, per esempio quello di incisi e di numerazione delle formule per le parentesi tonde, e quello delle correzioni e delle citazioni bibliografiche per le quadre. Sarebbe troppo confuso per il lettore distinguere più ruoli affidati alla stessa coppia di segni.

I trattini hanno ruoli diversi a seconda della loro lunghezza. Il trattino breve serve principalmente come indicatore di cesura in fin di riga; può svolgere anche il ruolo di trattino che unisce due parole associate nel loro significato cumulativo, ma tenute distinte per non formare una sola parola: *indo-europeo*. Va notato che quando il trattino unisce due locuzioni a loro volta formate da più di una

[7] Il latino ecclesiastico è latino moderno, non medievale; oltre ad avere uno stile francesizzante, usa gli accenti. Questo sembra sia richiesto per consentire ai fedeli e ai religiosi di leggere correttamente e all’unisono indipendentemente dall’accentazione che spontaneamente userebbero in base alla sensibilità ritmica derivata dalla loro lingua madre.

parola, il trattino viene separato con spazi insecabili dalla prima e dalla seconda locuzione: *Trentino - Alto Adige*.

Il trattino medio '–' (*en dash* in inglese) non deve essere confuso con il segno meno '−'; LaTeX mantiene anche graficamente un lunghezza diversa. Il segno meno si usa solo in matematica; se si deve indicare una quantità negativa espressa in cifre, la si componga in modo matematico, non si cerchi di simulare il segno meno con un trattino medio; "la borsa ha avuto variazioni giornaliere dal 2% al −3,5%". Il trattino medio viene usato specialmente per indicare intervalli numerici senza ricorrere alla composizione matematica che usa altri simboli più precisi; per esempio, nelle bibliografie, specialmente se si tratta di articoli, è opportuno indicare le pagine nelle quali essi si trovano stampati dentro al volume che li raccoglie; si scriverà dunque: "pag. 321–347".

Nei romanzi il trattino medio serve anche in sostituzione delle virgolette, per demarcare i dialoghi. In particolare una battuta generalmente inizia con un trattino medio e uno spazio; se essa finisce con il punto a capo, non ha bisogno di avere anche il trattino di chiusura. Se invece viene interrotto da una parte narrativa, o il testo prosegue sulla stessa riga con una parte narrativa, allora la battuta viene separata da quanto segue con il trattino medio preceduto da uno spazio. Nella letteratura anglosassone questo ruolo del trattino medio viene svolto dal trattino o lineato lungo.

Il trattino lungo (*em dash* in inglese) viene usato poco nella tipografia italiana; principalmente viene usato per indicare delle omissioni di lettere in una parola, per esempio per non svelare il nome di un personaggio si può scrivere: "il signor A—". Talvolta, imitando le tradizioni tipografiche angloamericane, il trattino lungo viene usato al posto del trattino medio.

Il segno di paragrafo '§' e di capoverso '¶' vengono usati raramente; il primo certamente viene usato più del secondo; questo a sua volta trova la sua collocazione specialmente nei libri liturgici insieme ad altri numerosi segni speciali; lo si vede anche nei libri che cercano di imitare la scrittura dei codici.

3.5 Abbreviazioni

Le abbreviazioni sono un altro terreno infido; spesso esse formano degli acronimi; in questo caso solitamente viene usata la lettera iniziale maiuscola (talvolta anche le due lettere iniziali) di ciascuna delle parole che formano il nome dell'ente o dell'entità di cui si usa l'acronimo. Alcuni acronimi sono diventati parole a sé stanti di cui si è perso nel comune sentire il significato della locuzione originale: per esempio, la parola comune *radar* è in realtà un acronimo, inizialmente scritto in lettere maiuscole RADAR, che sta per la locuzione inglese *RAdio Detection And Ranging*. L'acronimo CORECO sta per COmitato REgionale di COordinamento. L'acronimo ONU sta per Organizzazione delle Nazioni Unite; ovviamente in altre lingue, come l'inglese, l'acronimo corrispondente è UNO. Come fatto qui, gli acronimi sono composti con lettere maiuscole, preferibilmente in minuscole del maiuscoletto, senza punti di abbreviazione interposti.

Le abbreviazioni più comuni sono formate dalla prima sillaba e dalle prime consonanti della seconda sillaba e vengono terminate dal punto di abbreviazione. Raramente le abbreviazioni sono formate solo dalle prime consonanti della sola prima sillaba. In rari casi, al contrario l'abbreviazione è costituita dalle prime due sillabe della parola da abbreviare seguite dalle prime consonanti della terza sillaba e terminando il tutto con il punto di abbreviazione; per esempio 'teorema' può essere abbreviato in 'teor.'

Il burocratese ci ha inquinato con usi strampalati delle abbreviazioni, duplicando alcune lettere per indicare il plurale dalla parola abbreviata; questo avviene anche negli acronimi: VVF (Vigili del Fuoco), OO.SS. (Organizzazioni Sindacali), pagg. (pagine), sigg. (signori), proff. (professori). Sebbene il meccanismo sia chiaro, il plurale indicato in questo modo è pleonastico e fuorviante, visto che il significato plurale è implicito dal contesto.

In alcuni altri casi si sono sviluppate abbreviazioni stenografiche usando solo alcune consonanti della parola da abbreviare: btg. (battaglione), cfr. (confronta), e simili.

Altre volte vengono usate abbreviazioni quando non c'è nessun bisogno di abbreviare; quando si abbrevia 'sig.r' al posto di 'signor' si risparmia solo una battuta; lo stesso avviene per 'sig.ra' e 'sig.na'. Certamente queste abbreviazioni sono più adatte a una lettera commerciale che a un testo a stampa; il burocratese sia lasciato alla burocrazia.

Ma questi malvezzi si trovano anche negli scritti a stampa; per esempio in una bibliografia a piè di pagina può capitare di citare una nota e il numero di un volume; entrambi ammettono l'abbreviazione 'n.', ma è evidente che non si può usare la stessa abbreviazione per cose diverse; quindi è meglio non abbreviare piuttosto che abbreviare 'nota' in 'not.'; piuttosto si abbrevierà 'numero' in 'num.' riservando 'n.' alla nota, ma in entrambi i casi è meglio evitare l'abbreviazione, oppure, se proprio manca spazio, si può usare il simbolo № mediante il comando `\textnumero`.

Direi anche che le abbreviazioni del tipo F$^{\text{lli}}$, Ill$^{\text{mo}}$, Chiar$^{\text{mo}}$ con o senza sottolineatura dell'esponente, dovrebbero essere evitate in ogni modo, anche in burocratese, tranne nel caso in cui si debba scrivere la ragione sociale di un ente il cui "logo" comprende proprio questo genere di abbreviazioni.

Si ricorda infine che le abbreviazioni, eccettuati gli appellativi per i quali valga la pena, non si usano mai nel testo corrente, ma si possono usare solo negli incisi fra parentesi; si dirà dunque: "l'equazione 23 dimostra che…", e "(v. fig. 33)", ma non si dirà "l'eq. 23 dimostra che…" o, peggio ancora, "la 23 dimostra che…"[8].

[8] Tuttavia in scritti molto tecnici se il numero dell'equazione è racchiuso fra certe parentesi che vengono usate solo per indicare le equazioni, per esempio le parentesi tonde, allora si può omettere del tutto la parola "equazione" e si scriverà "la (23) dimostra che…".

3.6 Appellativi e maiuscole

In italiano le maiuscole si usano con grande parsimonia; purtroppo l'imitazione dell'inglese ci ha abituato a certe forme dove le maiuscole si sprecano.

Cominciamo dagli appellativi. Questi sono quei nomi che permettono di distinguere le persone o sul piano professionale, o sul piano istituzionale, come ingegnere, professore, re, presidente, eccetera. Anche il semplice 'signore' è un appellativo che la buona educazione fa premettere (nel genere e nel registro giusto) al nome di qualunque persona. Gli appellativi non vanno scritti con l'iniziale maiuscola quando sono seguiti dal nome della persona; quindi diremo semplicemente "l'ingegner Rossi", "il professor Bruni", "il re Gustavo", "il presidente Napolitano". Useremo l'appellativo con la maiuscola quando l'appellativo non è seguito dal nome proprio e si riferisce alla posizione istituzionale: "il Re", "il Presidente", "ho parlato con il Direttore", eccetera. Penso che l'ultimo esempio sia già un po' tirato per i capelli; personalmente scriverei "ho parlato con il direttore". Tuttavia si prenda in considerazione che la Gazzetta Ufficiale generalmente scrive "il Presidente della repubblica" lasciando l'iniziale minuscola a "Repubblica".

In qualche manuale di indicazioni editoriali si trova scritto che i nomi degli enti si scrivono con le iniziali maiuscole se sono locuzioni formate solo da una o due parole; in tutti gli altri casi si scrive con la maiuscola solo la prima parola; apparentemente le istruzioni editoriali della Gazzetta Ufficiale seguono questa regola. Siccome l'articolo fa parte integrante dei titoli di alcuni giornali, bisogna scrivere "La Stampa", "Il Giorno", ma sempre secondo questa regola bisognerebbe scrivere "Il resto del carlino".

Certamente in italiano non vanno scritte con la maiuscola le parole che indicano nomi comuni: figura, tabella equazione, teorema, ..., nemmeno quando vi si fa riferimento con il numero identificativo. Siccome poi la "figura 33" è un oggetto ben determinato, la locuzione "figura 33" deve essere usata con l'articolo determinativo o con la preposizione articolata. In nessun caso introducendo iniziali maiuscole. Così vale per tutti gli altri oggetti numerati o numerabili.

I titoli degli articoli o dei libri hanno in italiano la sola prima parola con l'iniziale maiuscola; i titoli degli articoli vanno sempre citati fra virgolette preferibilmente alte; i titoli dei libri non vanno mai citati fra virgolette ma in corsivo.

L'iniziale maiuscola va usata anche per i nomi propri dei palazzi, dei monumenti, degli aerei, delle navi, dei treni, delle automobili.

Gli aggettivi di nazionalità vanno minuscoli; se sono personalizzati e riferiti ad una persona specifica allora possono avere l'iniziale maiuscola. Parlando degli olandesi si indicherà la parola con l'iniziale minuscola, ma parlando di Van Gogh si potrà dire "l'Olandese", allo stesso modo come quando si dice "il Nostro" o, volendo, "l'Artista". Si tratta di usare un qualche appellativo senza farlo seguire dal nome proprio, ma come detto sopra, è già tirato per i capelli usare l'appellativo maiuscolo quando *non* ci riferisca ad una carica istituzionale.

I nomi dei giorni e dei mesi vanno minuscoli; i nomi dei punti cardinali vanno minuscoli a meno che non vengano usati per indicare delle regioni geografiche; quindi si parlerà "del grande Nord", ma si dirà che "gli uccelli migrarono verso nord".

Per concludere; in italiano le maiuscole si usano con parsimonia e la correttezza tipografica italiana non vuole che si imitino gli anglo-americani, come non si imitano i tedeschi o altri modi di scrivere nazionali che usano le maiuscole con regole diverse dalle nostre. Nel dubbio ci si astenga dall'usare iniziali maiuscole.

3.7 Uso dei font

La scrittura tipografica consente di sfruttare le possibilità espressive dei font mediante l'uso sistematico di certe serie o di certe forme che, scrivendo a mano, non sono disponibili. Non è opportuno fare sfoggio di infiniti font, appartenenti a famiglie, serie, forme e corpi diversi. Bringhurst [8] afferma che si dovrebbe usare solo la forma tonda, quella corsiva e quella maiuscoletta e sempre e soltanto nella serie media; questa affermazione è ripresa da Serra in [52]. Il nero dovrebbe essere sempre evitato. A queste raccomandazioni si ispirano sia il pacchetto `ClassicThesis` sia le classi `suftesi` e `octavo`, e senz'altro questa raccomandazione è validissima per testi di tipo letterario. Per quelli di tipo tecnico-scientifico non si rifugge dal nero per i titoli, anche di sezioni di livello inferiore a quello del capitolo, ma lo si esclude nel corpo del testo.

Così si è fatto anche in questo testo che usa le impostazioni di default per la classe `book`, ma si è cercato di evitare ogni comando che impostasse la serie nera, specialmente quando si sono definiti i comandi che compaiono nel file `MacroGuida.sty` usati per certe composizioni particolari necessarie per questo testo. In particolare, rispetto alle precedenti versioni, si è abbandonato il carattere bastoncino nero per caratterizzare i nomi dei pacchetti, e lo si è sostituito con il carattere obliquo della famiglia a spaziatura variabile *variabile typewriter type* della collezione Latin Modern, facilmente distinguibile dal corrispondente font obliquo della famiglia a spaziatura fissa *typewriter type*.

Per evidenziare qualche cosa si usa preferibilmente il corsivo, mai la sottolineatura, pratica esclusivamente destinata alla scrittura con le vecchie macchine da scrivere e alla scrittura a mano. Per i nomi propri si può usare il maiuscoletto; questo viene fatto quasi sempre nelle bibliografie, come viene fatto anche in questo testo.

Come già detto, i titoli dei libri e delle opere teatrali vengono evidenziati in corsivo, mentre i titoli degli articoli pubblicati in periodici o riviste specializzate vengono evidenziati mettendoli fra virgolette. I nomi propri dei palazzi, monumenti, aerei, eccetera, vanno scritti in tondo.

Il corsivo si usa anche per scrivere parole in lingua straniera (latino compreso); se una parola è di origine straniera ma è entrata in modo stabile nel lessico italiano, la si può scrivere in tondo, ma in questo caso la si assoggetta alle regole grammaticali dell'italiano, in particolare la si lascia invariata al plurale: tennis,

golf, goal, file, software, eccetera. Se la si vuole mettere al plurale con le regole della lingua di origine, la si deve trattare come parola straniera, e quindi la si deve comporre in corsivo. Si scriverà, quindi, "i curriculum" oppure "i *curricula*", ma non si potrà scrivere "i curricula". D'altra parte perché curriculum dovrebbe poter essere declinato, mentre "lapis", "album", "iter", eccetera, non vengono declinati?

Va da sé che se si vogliono usare i plurali delle lingue di origine, è necessario, oltre all'uso del corsivo, anche la perfetta conoscenza della grammatica della lingua d'origine. Attenzione! Questo è un terreno estremamente sdrucciolevole, specialmente se le parole in questione terminano con vocali presenti anche nelle determinazioni delle parole italiane. I frutti *kaki* sono caratterizzati da un nome giapponese, quindi non si può scrivere, anche italianizzando con la sostituzione della 'k' con la grafia corrispondente italiana: *il caco, i cachi*; infatti la lingua giapponese non modifica le parole al plurale e *kaki* vale tanto per *un kaki* quanto per *tanti kaki*. D'altra parte a nessuno verrebbe in mente di trattare i *kiwi* come i *kaki*; si è mai sentito dire o visto scrivere "questo chivo è molto maturo"?

In matematica la scelta dei font è vincolata da norme molto precise, almeno per la varietà di matematica scritta dai fisici e dai tecnologi, perché ci sono delle norme ISO e delle norme CNR-UNI che ne prescrivono gli usi corretti. Si ripete qui che le norme UNI in Italia hanno valore di legge; qualunque documento con carattere legale, un contratto, un capitolato, una perizia, perde la sua validità se non vengono rispettate le norme; le conseguenze negative potrebbero essere molto, molto pesanti!

3.8 Le note

Le note di commento al testo possono essere di tre tipi: le note marginali, le note a piè di pagina, le note di fine capitolo o di fine documento.

Le ultime andrebbero evitate con cura; sono di disagevole lettura, qualunque funzione svolgano, perché sono lontane dal testo a cui si riferiscono.

Le note marginali possono costituire dei commenti puntuali che si riferiscono al testo che esse affiancano, purché siano sufficientemente brevi e possano trovare posto davvero vicino al testo a cui si riferiscono. Richiedono la presenza di margini larghi e la capacità di allineare la prima riga della nota alla riga del testo a cui si riferiscono. Vengono usate raramente nei testi tecnico-scientifici, ma nei testi di scienze umane vengono usate abbastanza spesso. Ricordiamoci che le edizioni critiche possono avere diversi apparati di note; se in calce ad ogni pagina ci fossero 5 apparati di note, ogni pagina sarebbe tremendamente complicata. Spostare un apparato di note nel margine può costituire una notevole semplificazione.

La tradizione anglo-americana vorrebbe che per le note a piè di pagina il richiamo sia costituito da un numero o una lettera o un simbolo ad esponente e lo stesso richiamo sia ripetuto in calce prima del testo della nota. Fin qui non c'è nessuna differenza con la tipografia italiana. Ma la stessa tradizione

anglo-americana vorrebbe che il richiamo nel testo segua l'eventuale punteggiatura dopo la parola a cui la nota si riferisce; si dovrebbe scrivere "amigdala,[9]" e non "amigdala[9],". Questa raccomandazione o tradizione non sembra avere altra ragione che l'estetica; alla luce di quanto è stato detto sulla funzione dell'interpunzione, apparirebbe più corretto scrivere il richiamo di nota prima del segno di punteggiatura, in quanto la nota si riferisce ad una parola che si trova prima di quel segno di punteggiatura, segno che, appunto, separa una parte sintattico-grammaticale da un'altra.

Tuttavia entrambe le collocazioni del richiamo hanno la loro ragione d'essere; il compositore sceglierà quella che gli sembra più adatta al tipo di documento che scrive, ed avrà cura di rimanere fedele a quella scelta per l'intero documento.

3.9 Conclusioni

Come si vede le regole da seguire in tipografia sono molte e sono molte anche le consuetudini tipografiche alternative che si possono seguire; qualunque compositore dovrebbe seguire come prima regola la coerenza. Per rispettare la coerenza compositiva egli si trova avvantaggiato se usa LaTeX, perché mediante la definizione di opportune macro è in grado di comporre molte cose nello stesso stile; se in un secondo tempo dovesse decidere di cambiare stile, gli basta cambiare la definizione della macro e automaticamente tutto il documento viene composto nel nuovo stile in modo corretto e coerente.

LaTeX, però, non basta; ci sono diverse situazioni, come quelle relative alla punteggiatura, che vanno usate con coerenza, ma che non possono essere codificate dentro opportune macro. È utile allora che il compositore, se non gli fosse già imposto uno stile della casa editrice per cui compone, si scrivesse autonomamente un manualetto di stile; bastano poche pagine, dove vengono riassunte tutte le regole scelte e, magari, dove sono indicati anche i nomi delle macro che realizzano tutte o alcune di quelle scelte. In un manualetto del genere si possono anche scrivere alcuni promemoria per la punteggiatura, per l'evidenziazione, per le virgolette, per i plurali, per le citazioni in lingua straniera, eccetera, in modo da non dover consultare un libro alla ricerca della regola che non si ricorda più.

Tempo fa chi scrive contribuì con diversi articoli o lemmi al *Dizionario di ingegneria* pubblicato dalla UTET. Fu un opera colossale, magistralmente coordinata dalla redazione editoriale, che dovette assemblare i contributi di centinaia di persone. La precedente edizione era stata ideata e coordinata da Eligio Perucca, professore di fisica sperimentale al Politecnico di Torino.

Eligio Perucca era stato anche il presidente e coordinatore di diverse commissioni o gruppi di lavoro dell'Ente Italiano di Unificazione (UNI) e si era reso conto della necessità di una simbologia unificata nelle scienze e nelle tecniche. Per il *Dizionario di ingegneria* aveva esaminato con cura i font di cui disponeva la UTET e aveva predisposto un manualetto di regole editoriali di molte pagine, non una cosetta da poco, dove oltre alle regole ortografiche indicate in questo capitolo aveva riportato un interminabile elenco di simboli per indicare virtualmente

qualunque grandezza fisica potesse entrare in un dizionario di questo genere. Ai tempi della prima edizione il Sistema Internazionale non era ancora stato approvato; in fisica regnava l'anarchia più generale fra sistemi 'cgs', 'cgsm', 'cgse', 'cgs Gauss', 'mksa razionalizzato', 'mksa non razionalizzato', ed altri ancora. Eligio Perucca aveva dispiegato una varietà di simboli e vi aveva associato una varietà di nomi in modo che il *Dizionario* potesse contemperare senza confusione tutte le grandezze che fino ad allora erano state chiamate con gli stessi nomi e indicate con gli stessi simboli, quando in realtà si trattava di cose diverse. I font usati ricorrevano ad ogni possibile serie e ad ogni possibile forma disponibile nella tipografia in cui sarebbe stato composto il *Dizionario*. Si usavano senza sovrapposizioni i font corsivi con grazie, i font senza grazie, i font gotici rotondi; sia l'alfabeto latino sia quello greco; tutti nelle varianti medie e nere.

Questo manuale editoriale fu usato anche per la seconda edizione; fu preziosissimo sia per gli autori, che allora non disponevano ancora dei PC, sia per i compositori, sia per la redazione. Per chi scrive quell'esperienza fu importantissima; in particolare gli insegnò l'importanza dell'ortografia tipografica. Chissà se è riuscito anche a metterla in pratica.

Capitolo 4

Installare il sistema TEX

L'installazione del sistema TEX può essere molto semplice o relativamente complicata; dipende da quali strumenti si usano e da come l'utente sa usare efficacemente il sistema operativo del proprio PC o laptop.

Si distinguono tre casi:

1. Sistemi operativi Windows da Windows XP in poi con processore a 32 o a 64 bit;

2. Sistemi operativi Linux con processore a 32 o a 64 bit;

3. Sistema operativo Mac OS X dalla versione 10.5 con processore PowerPC oppure Intel a 32 o a 64 bit. Non più aggiornate, sono disponibili ancora le versioni per sistemi operativi precedenti a quello di versione 10.5.

Qui si parlerà delle distribuzioni freeware; per le distribuzioni commerciali l'utente deve valutare bene il rapporto prestazioni/prezzo in relazione alle proprie esigenze. Probabilmente è preferibile cominciare con una distribuzione gratuita, per poi passare ad una commerciale se e quando si sarà constatata le reale necessità di quelle caratteristiche in più che la distribuzione commerciale offre rispetto a quelle gratuite.

Va notato che l'organizzazione degli utenti di TEX, TEX Users Group (TUG), offre la possibilità di scaricare dal suo sito `www.tug.org` l'immagine del disco TEX Live, che, comunque, viene inviato gratuitamente a tutti i suoi soci; questo disco, dati i tempi per il download, può essere anche ordinato alla sede di TUG e lo si può ottenere praticamente al costo della duplicazione del disco o poco più. Questo disco contiene le distribuzioni per tutti i principali sistemi operativi e per tutte le macchine più diffuse; quello che si esporrà qui di seguito può essere ritrovato in quel disco.

4.1 Installazione su macchine Windows da XP in poi

4.1.1 Installare MiKTEX

La forma più semplice per installare il sistema TEX su una macchina Windows che usi un sistema operativo da XP fino ai più recenti, è quella di procurarsi il CD di installazione ProTEXt; detto CD può anche essere scaricato come un grande file compresso da uno dei siti dove si trova tutto il materiale relativo al sistema TEX. Per esempio ci si può riferire al sito

`http://www.tug.org/protext/`

e alle istruzioni che vi sono contenute; cliccando sull'apposito link, si tratta di scaricare un grande file di più di 1 GiB, quindi è necessario disporre di una connessione molto veloce. Esiste anche una immagine da masterizzare su un CD, che può essere usata direttamente dal CD senza installare nulla, ma che consente di provare l'intero sistema.

Estratta l'immagine dal file autoestraente scaricato dal sito, l'installazione parte da sola mediante il solito espediente dell'`Autorun.inf`; in realtà basta cliccare gli opportuni bottoni che attivano tutte le operazioni necessarie alle installazioni di tutte le varie parti del pacchetto. Conviene leggere il file `protext-setup-en.pdf` per capire bene cosa fare per l'installazione.

In verità il disco autoeseguibile ProTEXt è solo un programma di installazione; il sistema TEX che viene installato è MiKTEX (attualmente la versione 2.9), di cui si parlerà diffusamente anche in seguito; il file di istruzioni per l'installazione suggerisce di installare la versione *basic*; in questo testo, indipendentemente dalla distribuzione, si consiglia, invece, di installare *sempre* la distribuzione completa.

Il programma di installazione configura l'installazione completa in modo che sia adatta a comporre in ognuna delle circa 80 lingue e loro varianti che oggi è capace di gestire; naturalmente è ragionevole pensare che non verranno usate tutte, sicuramente non nello stesso documento; tuttavia la versione *basic* non configura il sistema per comporre documenti italiani, quindi, non fosse che per questo, conviene installare la versione completa.

Si noti che questa installazione si occupa di installare anche TeXstudio, un ottimo *shell editor* interattivo per lavorare con il sistema TEX, e il visualizzatore di file PDF SumatraPDF, che è sincronizzabile con l'editor così da poter passare rapidamente dalla finestra dell'editor al punto corrispondente della finestra del visualizzatore e viceversa. Vengono installati anche programmi accessori quali ghostscript, ghostview, praticamente necessari con ogni installazione.

La distribuzione MiKTEX è specifica per le macchine Windows; per queste macchine esiste anche la distribuzione TEX Live, che è multipiattaforma; essa può essere installata al posto di MiKTEX, ma generalmente gli utenti Windows preferiscono installare la distribuzione MiKTEX, la cui gestione appare loro più semplice. Non si nasconde il fatto che il procedimento di installazione può

sembrare più semplice, e sotto molti aspetti lo è, ma i problemi nascono dopo: infatti se si accetta l'impostazione predefinita, viene installata una versione di base, cosa che è sempre sconsigliabile fare, a meno che non si disponga di poco spazio sul disco fisso; questo era un problema con desktop e laptop di altri tempi, perché quelli di oggi dispongono di dischi fissi talmente capaci che sembra impossibile riempirli completamente. Inoltre, come già accennato, la versione di base non è configurata per comporre testi in italiano, per cui l'utente, specialmente se è nuovo a MiKTEX, si trova in difficoltà. Poi talvolta non si riescono a scaricare i pacchetti mancanti dalla rete, non per difetto di MiKTEX, ma per impedimenti causati dalla rete. Insomma, il forum del gruppo italiano GUIT contiene quasi giornalmente domande di aiuto da parte di utenti di MiKTEX che non riescono a configurare correttamente la loro distribuzione. Perciò si prenda in considerazione la possibilità di non installare MiKTEX e si veda poco più avanti la descrizione della installazione di TEX Live su una macchina Windows.

Terminata l'installazione guidata, il sistema è pronto per l'uso; se si è scelto di installare la versione di base, bisogna configurare il sistema per l'uso anche della lingua italiana.

4.1.2 Installare TEX Live

L'installazione di TEX Live sulle macchine Windows avviene come per le macchine Linux; l'unica differenza che il file di installazione che bisogna scaricare dalla rete o che si trova nel DVD di installazione, è più grande, perché deve provvedere ad installare alcuni linguaggi, come per esempio Perl, di cui le macchine Windows non sono normalmente dotate; a parte questo dettaglio che è del tutto trasparente per l'utente, il resto procede esattamente come per le macchine Linux, a meno che non venga descritta qualche procedura particolare riferita a questa o a quella realizzazione particolare del sistema operativo Linux.

Esiste però una alternativa costituita dallo scaricare dal sito CTAN l'immagine ISO del DVD distribuito ai soci dei vari gruppi nazionali o internazionali; e poi installare direttamente da questa immagine.

Infatti per usare l'immagine del disco come se fosse il DVD vero e proprio, basta disporre della versione o del sistema operativo Windows 8; se si dispone di una versione precedente, da WinXP a Win7, bisogna prima installare un "montatore" di immagine ISO come se fosse il disco vero e proprio; chi scrive ha installato il programma WinCDEmu scaricandolo dal sito `http://wincdemu.sysprogs.org/`. Quello che viene scaricato è in realtà il pacchetto di setup del programma, lanciato il quale in pochi secondi il sistema operativo diventa in grado di aprire una immagine ISO come se fosse un CD o un DVD vero.

Prima dell'installazione di una nuova versione di TEX Live, avendone già installata in una versione precedente, conviene rimuovere quest'ultima prima di installare quella nuova. Nell'installazione su macchine Windows, fra le cartelle del sistema c'è la cartella `installer`, dentro la quale compare anche la procedura uninstl.bat; lanciando questa procedura della finestra "prompt dei comandi", viene eseguita la rimozione della versione esistente di TEX Live così da poter

eseguire un nuova installazione pulita; eventuali configurazioni locali o pacchetti e programmi non facenti parte della distribuzione vera e propria, precedentemente installati nell'albero di TEX Live radicato in `texmf-local`, vengono conservati e non vengono cancellati dall'esecuzione della procedura suddetta e restano disponibili anche per la nuova versione.

Supponendo di avere scaricato l'immagine `texlive2013-20130530.iso` cliccando su uno dei link della pagina internet `http://www.tug.org/texlive/acquire-iso.html`, basta eseguire un doppio click sul nome del file appena scaricato; questo non solo apre il file come se fosse un CD, ma parte anche automaticamente il programma di installazione; basta rispondere a poche domande nella finestra interattiva che appare, e il sistema si installa da solo. Va notato però che l'immagine ISO viene aggiornata una volta all'anno; nell'esempio riportato sopra si tratta del file che contiene la distribuzione TEX Live2013 creata il 30 maggio 2013. Tuttavia fra le parti installate esiste anche il programma Perl tlmgr che permette di aggiornare la distribuzione quando si vuole; si consiglia di farlo settimanalmente o, almeno, ogni 10 giorni. Per usare tlmgr efficientemente basta consultare il suo manuale di istruzioni che si apre dando dalla finestra "prompt dei comandi" il comando:

```
texdoc tlmgr
```

Complessivamente l'installazione dall'immagine ISO richiede un poco più di tempo, rispetto all'installazione via Web, ma è molto meno soggetta ai capricci della rete, che talvolta è penosamente lenta[1].

4.2 Installazione su Linux

Parlare di Linux come se fosse un sistema operativo unico e omogeneo è un po' azzardato; esistono centinaia di distribuzioni Linux ognuna con le sue particolarità; queste consistono in diversi "package manager", diversi schemi di controllo della validità e compatibilità dei vari moduli, eccetera. Inoltre le dipendenze reciproche dei vari moduli sono molto intricate e spesso diverse da una distribuzione Linux da un'altra.

TEX Live è disponibile anche per i sistemi Linux ma richiede alcune attenzioni, specialmente se la versione Linux su cui si deve installare il sistema TEX è soggetta ai vincoli del consorzio Debian.

In ogni caso quasi tutte le varianti Linux dispongono di loro "repositories" da cui scaricare i loro programmi, tra i quali il sistema TEX. Non se ne parla qui, perché perché le varianti sono troppo numerose, ma ogni utente Linux sa come usare i suoi "package manager" e quindi scendere nei dettagli è probabilmente eccessivo.

[1] Scaricare dai *mirror* italiani è spesso un'impresa: quello di Roma2 spesso è sovraccarico; quello della rete GARR funziona meglio; dall'Italia settentrionale è possibile e conveniente scaricare da siti stranieri svizzeri, austriaci, francesi o anche olandesi, che permettono velocità maggiori. Naturalmente è possibile scaricare da questi siti stranieri anche dall'Italia meridionale, ma le tratte più lunghe implicano anche un rallentamento delle operazioni.

Va però specificato che la versione di TEX Live convalidata dal consorzio Debian è solitamente un po' datata, in passato anche di un paio d'anni, oggi al massimo di un anno, anche perché non è aggiornabile con il suo programma tlmgr (TEX Live manager). Inoltre costituisce una dipendenza per diversi altri programmi. Esistono due alternative a questi problemi.

1. Si configura il sistema Linux in modo da fargli credere di avere già installato il sistema TEX; come farlo dipende dalla particolare installazione di Linux, quindi ogni utente deve documentarsi in merito facendo riferimento alla sua particolare distribuzione. Dopo questa operazione si può installare TEX Live agendo dal sito CTAN come indicato per il sistema operativo Windows.

2. Si lascia che i programmi che dipendono dalle funzionalità di TEX Live/Debian vi facciano riferimento in modo normale, ma accanto alla installazione conforme alle direttive Debian, ridotta al minimo indispensabile, si installa la versione TEX Live/CTAN completa. Le due versioni possono convivere tranquillamente una accanto all'altra se la seconda versione viene installata seguendo le indicazioni riportate nell'articolo [24][2]. Quindi la prima versione TEX Live/Debian serve per le dipendenze e la seconda versione TEX Live/CTAN serve per usare il sistema TEX sempre aggiornato o aggiornabile a piacere.

In entrambi i casi sui sistemi Linux sono disponibili diversi *shell editor* adatti per gestire i documenti da comporre con il sistema TEX; Kile è un programma Debian-compatibile e funziona solo con la versione TEX Live/Debian; emacs è sempre presente con i sistemi Linux: arricchito con il plugin `auctex` va benissimo per gestire i documenti da comporre con il sistema TEX. Sono però disponibili altri *shell editor* multipiattaforma da TeXworks a TeXStudio e TeXmaker che sono particolarmente comodi perché sono già sincronizzati con un loro visualizzatore di fie PDF interno che permette di eseguire la ricerca diretta e inversa passando dalla finestra di editing al punto corrispondente della finestra PDF e viceversa.

4.3 Installazione su Macintosh con Mac OS X

Da quando esiste il sistema operativo Mac OS X e la macchina a sua volta usa un processore Intel Dual Core o superiore, oppure PowerPC, esiste la versione universale di MacTEX, valida per entrambe le macchine dotate di quei processori; per la verità recentemente la distribuzione aggiornata serve per l'installazione solo su macchine dotate di sistema operativo con un numero di versione 10.5 o superiore; per le vecchie macchine con sistemi operativi della serie 10, ma precedenti alla 10.5, si possono scaricare dalla rete le "vecchie" versioni, che però non sono più aggiornate, benché siano ancora disponibili. Senza voler spingere

[2]Questo articolo descrive come installare TEX Live sulla versione Ubuntu/Linux, ma contiene anche le piccole varianti per eseguire la stessa installazione sui sistemi Fedora e OpenSuse.

a spese non previste, sarebbe meglio in realtà aggiornare il sistema operativo oppure acquisire una macchina più moderna.

Il file di installazione è[3]:

`http://tug.org/mactex/`

dove seguendo le istruzioni, si può scaricare il pacchetto *MacTeX.pkg* specifico per le macchine Macintosh; scaricato questo pacchetto, basta cliccarci sopra e si apre il solito dialogo di installazione; conviene rispondere che si vuole l'installazione completa. Il programma provvede da solo ad eseguire l'installazione completa di una distribuzione TEX Live adattata al sistema di cartelle dei calcolatori Macintosh.

Essa installa anche gli *shell editor* TEXShop e TeXworks; l'utente che trovi più comodi altri editor deve provvedere separatamente; si veda più avanti a questo proposito. Invece i due programmi citati sopra sono già dotati delle funzionalità necessarie per visualizzare il risultato della composizione e per procedere alla stampa. Infatti, il visualizzatore della composizione è integrato sia in TEXShop sia in TeXworks ed è già configurato per passare puntualmente dal file sorgente al punto corrispondente del file PDF compilato o viceversa; questa funzionalità agevola enormemente la lavorazione del documento che si sta scrivendo.

Viene installata anche l'applicazione TeX Live Utility in `Applications/TeX/`; essa è una interfaccia grafica in stile Macintosh per operare con il TEX Live manager; molto comodo e intuitivo per eseguire gli aggiornamenti.

Vengono installati anche i programmi BibDesk e LaTeXiT; il primo serve per gestire i database bibliografici; il secondo serve per comporre con latex le formule matematiche eventualmente utili anche sotto forma di immagine, per inserirle dentro altri formati di documenti. Volendo la si potrebbe introdurre anche nei documenti da produrre con latex e pdflatex, ma sarebbe veramente stupido inserire una immagine composta tramite LaTeXiT dentro un documento da comporre con uno dei programmi del sistema TEX, quando questo programma sarebbe capace di produrre la formula da solo e senza i problemi di compatibilità del contenuto che si verrebbero a creare.

Al massimo LaTeXiT, a parte gli altri documenti composti con programmi e in formati diversi da quelli che si possono comporre e produrre con i programmi del sistema TEX, può servire come ambiente interattivo per verificare la correttezza della formula via via che la si compone; poi finita la composizione, si copia il codice sorgente e lo si incolla nel file `.tex` del documento da comporre.

[3] Oggi sovente ci si riferisce ad indirizzi come `mirror.ctan.org/...` invece che ad indirizzi espliciti come quello indicato. Riferirsi ad un mirror è una operazione intelligente, perché si impegna la rete per un tempo minore e generalmente su un numero minore di tratte; il guaio per noi italiani è che, riferendosi alla lista di mirror gestita automaticamente dal software di installazione, spesso si è indirizzati al mirror di Roma2, noto per la sua lentezza e per la frequenza con la quale si manifestano interruzioni di servizio. Il mirror GARR di Bologna è certamente più affidabile; tuttavia, almeno per gli utenti dell'Italia settentrionale è spesso conveniente riferirsi all'Università Tecnica di Vienna, oppure al mirror olandese UCS, oppure all'INRIA francese. Altrove si è già discusso dei mirror utili per gli utenti italiani.

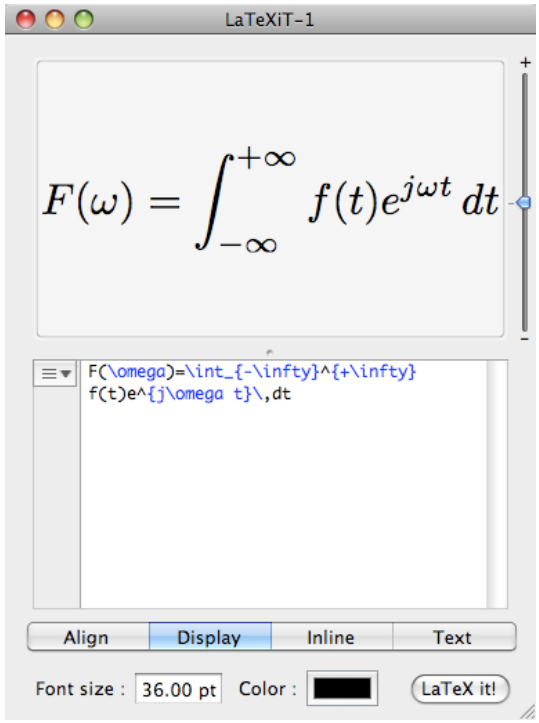


Figura 4.1: Schermata di LaTeXiT

Se questo discorso può apparire confuso, si osservi la figura 4.1; essa mostra una finestra di lavoro del programma LaTeXiT divisa in due parti; nella parte inferiore si scrive la formula in linguaggio LaTeX standard, senza cioè estensioni di nessun genere, semplicemente usando la tastiera, non il mouse; nella parte superiore, dopo aver cliccato sul bottone in basso a destra, etichettato con il nome LaTeXiT! appaiono i messaggi di errore, casomai ne fossero presenti, oppure l'immagine della formula composta. Se la si vuole salvare come immagine, nella barra del menù a discesa di File compare la dizione Export image scegliendo la quale si salva il disegno della formula nel formato vettoriale PDF con il nome che gli si specifica.

Lavorando con il sistema TeX, conviene invece selezionare dalla finestra inferiore il codice, copiarlo e incollarlo nel file `.tex` del documento che si sta componendo. Se si vogliono eseguire modifiche per usare comandi non standard o ambienti diversi da quelli standard, basta correggere il codice copiato. Siccome scrivere direttamente in linguaggio LaTeX è una cosa che diventa via via più facile man mano che si acquisisce esperienza, ci si accorge in breve tempo che LaTeXiT non è più necessario (quando si compone usando uno dei programmi del sistema TeX) e non lo si usa più o lo si usa solo saltuariamente. Per questo motivo non se ne parlerà più nel seguito.

4.4 Gli alberi di cartelle del sistema TEX

L'installazione dell'uno o dell'altro sistema crea sul disco diverse strutture di cartelle; ovviamente queste strutture di cartelle, gli *alberi* di cartelle, possono essere un po' diversi da sistema operativo a sistema operativo, da installazione a installazione.

La differenza più ovvia è che nella descrizione del percorso sulle macchine Windows il separatore fra i nomi delle cartelle è una barra rovescia, \, mentre sulle macchine Linux e Mac OS X il separatore è una barra semplice, /. In questo testo si userà sempre la barra semplice per due motivi: (*a*) la barra rovescia ha un significato speciale per il linguaggio LATEX, per cui limitarne l'uso alla sua funzione speciale aiuta anche a evitare errori nello scrivere il testo; (*b*) per lo stesso motivo scrivendo in linguaggio LATEX, anche su una macchina Windows, i percorsi dei file devono essere sempre indicati con la barra semplice invece che con la barra rovescia.

Le differenze più sostanziali riguardano l'installazione TEX Live rispetto a quella MiKTEX. Se ne parlerà più diffusamente più avanti. Qui ci si limiterà ad esporre le similitudini.

Tuttavia l'argomento può sembrare un po' ostico; in un certo senso lo è, ma il lettore non si scoraggi: salti direttamente al paragrafo 4.5. Può tornare sopra questo argomento in seguito, quando vorrà approfondirlo.

Tutti gli alberi, a partire dalla loro 'base' o 'radice', devono essere conformi alla *TEX Directory Structure* (TDS), cosicché il sistema TEX possa trovare i file che gli servono sempre nello stesso modo e con un unico suo programma interno di ricerca dei file che si chiama kpsewhich; questo programma può anche venire usato dall'utente, ma di solito egli ne ignora persino la presenza perché, in effetti, esso svolge le sue funzioni dietro le quinte ed è molto raro che l'utente debba servirsene direttamente. La TDS è descritta nel file `tds.pdf` in grande dettaglio; non è necessario che l'utente fin dalla prima installazione si studi approfonditamente questo documento, ma è importante che ne conosca l'esistenza per poterlo consultare quando occorre. Esso viene installato insieme al sistema e può essere letto aprendo la finestra comandi, console, xterm, Terminal, comunque si chiami quella finestra nel particolare sistema operativo in uso, e scrivendoci dentro il comando:

```
texdoc tds
```

Il comando texdoc è utilissimo anche per consultare la documentazione di qualsiasi pacchetto; solitamente basta dargli in pasto il nome del pacchetto e questo programmino esegue le ricerche necessarie per trovarne la documentazione e presentarla sullo schermo.

Ciò premesso, ogni utente farebbe bene a costruirsi un suo albero personale, conforme alla TDS, dove conservare le sue macro, i font che egli si crea, i pacchetti che si scrive; insomma tutto quanto gli sia strettamente personale e sia funzionale al funzionamento dei programmi del sistema TEX. Questo albero personale parte dalla cartella `texmf` che si trova nella sua `$HOME`; questo nome simbolico `$HOME` ha

un significato diverso in ogni sistema operativo e deve essere diverso dalla 'radice' sulla quale sono innestati gli alberi installati dal programma di installazione. Normalmente questo nome simbolico rappresenta:

- `~/` nei sistemi Linux;
- `~/Library/` nei sistemi Mac OS X;
- `C:\Documents and Settings\`⟨*utente*⟩`\Documenti\` con l'installazione di MiKTEX sulle macchine Windows XP; sulle versioni Windowds Vista, Seven, Eight, vale `C:\Users\`⟨*utente*⟩`\Documenti\`;

dove ⟨*utente*⟩ è il nome con il quale il particolare utente si è registrato sulla macchina.

Non si parlerà più di questo albero personale se non riferendosi ad esso con il generico nome di *albero personale* che, appunto, in quanto tale, verosimilmente non avrà l'intera struttura TDS, ma solo, conformemente a questa struttura, le cartelle di cui il particolare utente effettivamente necessita.

Ogni programma eseguibile del sistema TEX sa dove cercare i vari tipi di file di cui ha bisogno; per ogni tipo di file può eseguire la ricerca in diversi alberi, ma sempre con una precedenza: gli alberi prioritari sono quelli dell'utente specifico, poi vengono quelli della macchina specifica, infine quelli generali dell'installazione. Siccome si tratta di alberi che contengono migliaia di file, la base di ogni albero contiene un database dei nomi dei file che di solito deve venire aggiornato ad ogni modifica. Solitamente non vengono modificati a mano gli alberi del sistema TEX, ma l'albero personale è lì apposta per ricevere ogni file personale dell'utente, perciò questo file viene modificato relativamente spesso. Ogni installazione dispone di comandi specifici per l'aggiornamento dei database dei nomi dei file; MiKTEX dispone addirittura di un *wizard* che richiede solo di cliccare su un bottone di una finestra di interfaccia grafica. In ogni caso i programmi che modificano, aggiungono o tolgono file dagli alberi di sistema provvedono da soli ad aggiornare gli specifici database dei nomi dei file. *Perciò l'utente deve ricordarsi sempre di aggiornare questo database del suo albero personale*, solo se ha installato la distribuzione MiKTEX; la distribuzione TEX Live, su tutte le macchine, non richiede l'aggiornamento del database dei nomi dei file dell'albero personale.

Si noti con attenzione: l'albero personale, proprio perché personale, non viene creato dal programma di installazione della particolare distribuzione che viene installata sulla macchina; se l'utente ne ha bisogno, deve crearselo da solo; gli si consiglia di rispettare le direttive della TDS, magari limitandosi a creare solo quelle parti della struttura TDS che effettivamente gli servono. Le diramazioni particolari che sarebbe bene creare nell'albero personale sono formate dalle cartelle: `fonts/`, `doc/ source/`, `tex/`, e in queste cartelle verrano via via aggiunte altre cartelle (sempre secondo la TDS) man mano che ne sorge la necessità.

4.4.1 Gli alberi di cartelle di TEX Live

Gli alberi che vengono gestiti più o meno direttamente dal sistema TEX e nelle cui cartelle l'utente in generale non può scrivere nulla, se non operando da "amministratore" o da "root" o da "superuser", con una distribuzione TEX Live sono innestati tutti in ⟨*radice*⟩`/texlive/`. Essi sono:

1. ⟨*radice*⟩`/texlive//`⟨*anno*⟩`/texmf-dist` contiene tutti i file che servono per il funzionamento del sistema TEX, tranne gli eseguibili veri e propri che sono conservati in un'altra cartella.

2. ⟨*radice*⟩`/texlive//`⟨*anno*⟩`/texmf/` contiene tutti i file necessari per la configurazione del sistema TEX generati al momento dell'installazione del sistema TEX; sono quindi file che riguardano la configurazione e il funzionamento del sistema TEX ma sono validi per tutti gli utenti della particolare macchina sul cui disco appare questo albero Dalla distribuzione 2013 questo albero è stato fatto confluire nel precedente.

3. ⟨*radice*⟩`/texlive//`⟨*anno*⟩`/texmf-config` contiene tutti i file di configurazione di default del sistema TEX; in altre parole sono i file che servono per configurare il sistema prima che eventualmente l'utente specifico o l'amministratore della macchina non decidano di impostare una configurazione particolare, non standard.

4. ⟨*radice*⟩`/texlive//`⟨*anno*⟩`/texmf-var` contiene tutti i file di configurazione del sistema TEX generati in seguito a comandi specifici dell'utente amministratore per una configurazione ad hoc dell'installazione sulla specifica macchina.

5. ⟨*radice*⟩`/texlive//texmf-local` contiene tutti i file utili all'esecuzione dei programmi del sistema TEX e che sono specifici della particolare macchina; in particolare esso contiene certi file di configurazione che dipendono solo da quanto è disponibile sulla macchina specifica.

6. ⟨*radice*⟩`/texlive//`⟨*anno*⟩`/bin` contiene tutti i programmi eseguibili del sistema TEX; il percorso che conduce a questa cartella e alle cartelle che esso contiene viene aggiunto automaticamente alla variabile di sistema `PATH` in modo che i programmi contenuti in questo particolare albero possano venire eseguiti sia dall'utente mediante comandi manuali, sia dagli altri programmi del sistema.

7. `$HOME/texlive` contiene un albero con la sua radice che comincia a sua volta con ⟨*anno*⟩ e prosegue con una TDS contenente i file di servizio e di configurazione generati dall'utente specifico (non in qualità di amministratore) e dai programmi di servizio durante l'esecuzione di composizioni per quello specifico utente; tra gli altri vengono conservati qui i file relativi ai font a matrici di punti che il sistema genera attraverso il programma

METAFONT quando ne ha bisogno o quando manca di quei font nel formato vettoriale (che sarebbe sempre il formato preferito e da preferire); qui vengono conservati anche i file metrici relativi a questi font generati con METAFONT. Poiché questa cartella ha la radice nella `$HOME` dell'utente, i file contenuti in questo albero sono specifici per un solo utente, il proprietario della cartella `$HOME`, ma non deve confondere questo albero con l'albero personale radicato in `$HOME/texmf`, di cui si è già parlato.

L'indicazione ⟨*radice*⟩ rappresenta il percorso che va dalla radice *del disco* su cui è installato il sistema TEX fino alla radice *dell'albero* che ci interessa; a seconda del sistema operativo potrà essere una cosa come `C:\`, oppure `/usr/share/`, oppure `/usr/local/`. Inoltre la doppia barra `//` dice al sistema TEX che quel percorso non si limita alle cartelle esplicitamente indicate, ma prosegue per tutte le ramificazione dell'albero fino alle ultime "foglie".

Vale la pena di ricordare la necessità di avere per ciascun albero il database dei nomi dei file sempre aggiornato; aggiornarlo quando non è necessario non produce nessun danno, ma omettere l'aggiornamento può costituire una fonte di frustrazione non indifferente perché il sistema potrebbe non essere in grado di trovare i file di cui necessita. Si ripete: *questo aggiornamento è particolarmente importante per l'albero personale con l'installazione MiKTEX.*

Tutto ciò può apparire estremamente complicato; in realtà l'utente o l'amministratore non hanno bisogno di occuparsene se non in circostanze eccezionali; è opportuno però conoscere l'esistenza di questa molteplicità di alberi e il loro contenuto. Una cosa importantissima da ritenere è che ad ogni installazione di un aggiornamento del sistema TEX vengono riscritti alcuni o tutti i file contenuti nelle cartelle la cui radice sia formata dalla sequenza di cartelle `texlive/`⟨*anno*⟩. Qualunque personalizzazione, quindi, non deve venire scritta dentro cartelle che si trovino dentro questi alberi, altrimenti se ne perde tutto il contenuto, perché verrebbe sovrascritta dai file di aggiornamento. Quando si installa una nuova versione, cambia l'⟨*anno*⟩ e quindi cambia tutta l'installazione; su alcuni sistemi è possibile far convivere diverse versioni del sistema TEX, ma questa pratica è consigliabile solo per utenti espertissimi che abbiano esigenze molto particolari.

4.4.2 Gli alberi di cartelle di MiKTEX

Sulle macchine Windows è possibile installare anche la distribuzione TEX Live del sistema TEX, ma spesso gli utenti delle macchine Windows installano la distribuzione MiKTEX.

I diversi sistemi operativi che vengono usati su queste piattaforme Windows sono molto variabili e non sempre completamente compatibili. Qui si farà riferimento ad una installazione con il sistema operativo Windows XP, che è ancora quello di prima installazione su diverse piccole macchine portatili; quanto si dice qui non dovrebbe essere sostanzialmente diverso da quello che potrebbe aversi su altri sistemi operativi più recenti come Windows Vista o Windows 7 o Windows 8.

La moltitudine di alberi che si sono visti con la distribuzione TEX Live si ripete anche con MiKTEX, sebbene con un numero minore di alberi. Qui di seguito si indicherà con ⟨*radice*⟩ il percorso dalla radice *del disco* `X:\` *alla radice* dell'albero specifico che verrà descritto. I programmi eseguibili e della prima installazione non saranno necessariamente installati nel disco `C` e per questo qui si usa la lettera `X` per designare il disco su cui è installato il sistema TEX, anche se quasi sempre `X` coincide con `C`. In generale su XP si ha:

$$\langle radice \rangle = \texttt{C:\Documents and Settings}$$

mentre su Windows Vista e sulle versioni successive del sistema operativo si ha:

$$\langle radice \rangle = \texttt{C:\Users}$$

Si noti che nel seguito ⟨*versione*⟩ indica la versione della distribuzione MiKTEX, per esempio `2.9`.

1. `X:\Programmi\MiKTeX`⟨*versione*⟩`\` contiene tutti gli eseguibili del sistema TEX nella sottocartella `miktex\bin\`, ma contiene anche tutti i file che nella versione TEX Live sono contenuti negli alberi radicati in `textmf-dist` e `texmf-config`; in particolare i file di configurazione di default, i file di documentazione, e tutti gli altri file necessari per il funzionamento completo del sistema TEX.

2. ⟨*radice*⟩`\All Users\Dati Applicazioni\MiKTeX\`⟨*versione*⟩`\` contiene la struttura di cartelle contenenti i file generati dai programmi di configurazione e dai programmi di generazione dei font validi per tutti gli utenti della specifica macchina.

3. ⟨*radice*⟩`\`⟨*utente*⟩`\Impostazioni locali\Dati Applicazioni\MiKTeX \`⟨*versione*⟩`\` contiene i file di configurazione e quelli relativi ai font relativi all'uso dell'utente specifico.

4. ⟨*radice*⟩`\`⟨*utente*⟩`\Dati Applicazioni\MiKTeX\`⟨*versione*⟩`\` contiene i file di prima installazione e quant'altro sia necessario per il funzionamento del sistema TEX per l'utente specifico.

5. ⟨*radice*⟩`\`⟨*utente*⟩`\Documenti\` contiene l'albero personale di cui si è parlato precedentemente.

Non c'è una completa corrispondenza fra gli alberi di MiKTEX e quelli di TEX Live, ma non è difficile capire la strategia che sta sotto questa apparente complicazione; il documento `tds.pdf` spiega questa strategia illustrandone anche i pregi e i difetti, nonché i compromessi che è stato necessario raggiungere per avere un sistema funzionale e realizzabile.

Anche con MiKTEX l'utente non deve preoccuparsi più di tanto, se non nei rari casi in cui debba cercare un file specifico. Ricordi inoltre che il 'wizard' di MiKTEX per configurare il sistema, aggiornarlo, arricchirlo di altri pacchetti, eventualmente

anche *on demand*[4], permette a MiKTEX di avere quelle funzionalità che lo rendono particolarmente gradito agli utenti dei sistemi operativi Windows.

4.4.3 L'aggiornamento dei database dei nomi dei file

L'argomento è talmente importante che si vogliono qui riassumere e ripetere le informazioni sparse nei paragrafi precedenti, anche per sottolineare questa questione cruciale.

Fintanto che si installa il sistema TEX, lo si configura, si installano nuovi pacchetti o se ne rimuovono alcuni di quelli già installati, si aggiornano sia gli eseguibili sia i pacchetti, ma sempre facendo uso del programma MiKTeX Settings o MiKTeX Maintainance per l'installazione MiKTEX per Windows, o TeX Live Manager per l'installazione TEX Live per tutti i sistemi, non è il caso di aggiornare il database dei nomi dei file, perché ci pensano quei programmi.

Va notato che aprendo MiKTeX Settings dal menù a tendina che viene attivato selezionando successivamente `Start|Programmi|MiKTeX`, si hanno due opzioni di 'manutenzione' (Maintainance): una da amministratore e una da utente. La prima serve per aggiornare i file di sistema e quelli a disposizione di tutti gli utenti della macchina, mentre il secondo serve per aggiornare solo i file a disposizione dell'utente specifico che sta usando il programma di manutenzione.

Il programma di manutenzione e di gestione dell'installazione TEX Live, tlmgr (che può operare sia in modo testuale dalla finestra comandi, sia in modo grafico interattivo), deve sempre venire usato con le prerogative dell'amministratore e, quando si attiva la funzione di aggiornamento dei database dei nomi dei file, li può aggiornare tutti cliccando un solo bottone.

Di solito non è necessario ricorrere a questo aggiornamento quando le modifiche ai file di sistema sono state eseguite attraverso i rispettivi 'manager'. Ma quando non è necessario aggiornare questi database oltre ai casi gestiti direttamente dal 'manager'? Tranne in un solo caso, quello relativo all'albero personale solo quando si sia installato a la distribuzione TEX Live, altrimenti *bisogna sempre aggiornare i propri alberi personali e gli alberi di sistema, se e quando si fossero installati dei pacchetti senza ricorrere ai 'manager'.*

Riepilogando, quindi, quanto detto sopra, bisogna sempre aggiornare i database dei nomi dei file nei casi seguenti:

[4] *On demand* non significa alla lettera 'a richiesta'. La cosa è molto più sottile: si immagini che l'utente stia compilando un documento che richieda l'uso di un certo pacchetto di estensione; l'utente lancia il programma di composizione, ma quando questo cerca di caricare quel pacchetto e non lo trova , controlla che faccia parte dei pacchetti disponibili per MiKTEX, ma non sia stato ancora installato; allora il programma di composizione si ferma, invoca il 'wizard' (il mago, lo stregone), che chiede all'utente se vuol installare il pacchetto mancante; alla riposta affermativa dell'utente il 'wizard' cerca in rete l'installatore di quel pacchetto, provvede all'installazione, all'aggiornamento dei file di configurazione, all'aggiornamento dei database dei nomi dei file e, quando ha terminato, restituisce il controllo al programma di composizione che prosegue usando anche il pacchetto appena installato. Non è una funzionalità da poco! Peccato che non funzioni se non si ha accesso alla rete; per questo e altri motivi si consiglia sempre di eseguire una installazione completa.

1. Quando si aggiungono o si tolgono dei file dal proprio albero personale (installazione TEX Live esclusa).

2. Quando si aggiungono o si tolgono file dagli alberi che contengono informazioni per l'uso locale del sistema TEX.

3. Quando si installano negli alberi di sistema pacchetti di estensione o pacchetti di font senza usare i dovuti 'manager'.

L'ultimo caso non dovrebbe presentarsi mai, ma talvolta si ha a che fare con pacchetti sperimentali, non ancora previsti per l'installazione ufficiale; sarebbe preferibile inserirli nel proprio albero personale, ma talvolta (raramente) si rende necessaria l'installazione globale.

Ma negli altri casi *bisogna sempre aggiornare il database dei nomi dei file.* Non farlo metterebbe il sistema TEX nell'impossibilità di trovare i pacchetti che gli servono, e quindi verrebbero emessi messaggi d'errore la cui comprensione sembra essere difficoltosa, a quanto si deduce dalla lettura dei messaggi inviati al Forum del gruppo GUIT.

4.5 I programmi accessori

La composizione differita consente di usare un programma specializzato per ciascuna operazione di composizione, in modo da usare sempre il prodotto migliore per ciascuna fase di 'lavorazione' del documento.

I programmi accessori più importanti sono i seguenti:

1. Lo *shell editor* serve per generare e/o modificare i file sorgente da dare in pasto a LATEX. Se ne discuterà diffusamente ne prossimo paragrafo.

2. Benché il formato 'standard' DVI (DeVice Independent) del file di uscita del sistema TEX oggi sia usato molto raramente, esso era e rimane il formato di uscita più o meno predefinito; perciò è necessario disporre del programma specializzato per rendere il frutto della composizione visibile e leggibile sullo schermo, o per renderlo stampabile con una varietà di dispositivi diversi, che possono andare dalle stampanti alle fotocompositrici. Ogni sistema TEX è corredato di un programma per visualizzare i file DVI, spesso utile anche per stampare. Talvolta è necessario convertire il file dal formato DVI al formato PostScript, e per gestire questo formato funziona egregiamente il programma dvips; ma per visualizzare il file così prodotto è utile disporre del programma ghostview, meglio conosciuto come gview o anche come gv; questo programma è una comoda interfaccia grafica per maneggiare un file PostScript al fine di visualizzarne il contenuto logico, per stamparlo e anche per trasformarlo in formato PDF. Chi esegue il lavoro vero è ghostscript, ma gview rende molto più agevoli tutte le operazioni, perché *ghostscript* è un poderoso programma, che però richiede di essere azionato dalla linea di comando (dalla finestra comandi di Windows o dalla

finestra Terminal, Xterm, o Console dei sistemi UNIX, Linux e Mac OS X). Generalmente gli *shell editor* dispongono di un bottone sia per azionare il programma di visualizzazione per il formato DVI, sia per lanciare dvips, sia per visualizzare il file PS con il programma gview. Tutte queste operazioni oggi sono eseguite raramente, ma il programmi accessori per eseguirle fanno già perte di ogni distribuzione o sono annessi ad ogni distribuzione del sistema TEX.

3. I file in formato PDF, oggi preferito rispetto al formato DVI, sono maneggiabili mediante gli appositi programmi caratteristici di ogni sistema operativo; per tutti la Adobe produce il programma gratuito Adobe Reader, che consente di visualizzare i file nel formato PDF e di ottenerne alcune informazioni. Se si desidera la piena funzionalità sarebbe necessario acquistare da Adobe il programma Adobe Acrobat (eventualmente nella versione professional), oppure altri programmi simili; Adobe Acrobat consente di eseguire qualche ritocco al contenuto del file, di estrarne delle intere pagine o delle figure, di eseguire il ritaglio di parti della pagina, ovvero di scontornare le immagini; inoltre consente altre funzioni, in particolare la verifica della conformità di un file PDF alle norme relative all'archiviazione del documento elettronico; di questo si parlerà più diffusamente nel capitolo 22.

   Per le macchine Mac oltre a Adobe Reader (o Adobe Acrobat) il visualizzatore di sistema si chiama Preview (o Anteprima se il sistema operativo è localizzato in italiano); vale la pena di installare anche il visualizzatore Skim scaricandolo dal suo sito `http://skim-app.sourceforge.net/`; esso ha le prestazioni di Preview arricchite di molte altre utilissime funzioni, compresa quella di essere sincronizzabile con alcuni *shell editor*.

   Incidentalmente *sincronizzabile* è una espressione che può indurre a malintesi, ma è quella che viene usata per descrivere la funzione per la quale si può passare dalla finestra di editing del file sorgente al punto corrispondente della finestra di visualizzazione del file composto e viceversa. Questa funzione è fondamentale per facilitare la composizione di un documento; generalmente essa è presente fra la finestra di editing e la finestra di visualizzazione del file composto in formato DVI; solo pochi editor e visualizzatori consentono la sincronizzazione se il visualizzatore serve per i file composti in formato PDF.

4. Sono utili anche diversi altri programmi di conversione di formato, non necessariamente già in dotazione del sistema operativo in uso, ma che tornano utili in diverse circostanze; per esempio jpegtops trasforma le immagini dal formato JPG (o JPEG) al formato PostScript, o meglio, Encapsulated PostScript. Il programma ps2pdf trasforma i file dal formato Encapsulated PostScript al formato PDF. Il programma gimp, nato per le piattaforme UNIX/Linux e ora disponibile anche per le altre piattaforme Windows e Mac, serve per modificare le immagini di quasi tutti i possibili formati e di salvarle in quasi tutti gli innumerevoli formati che è capace

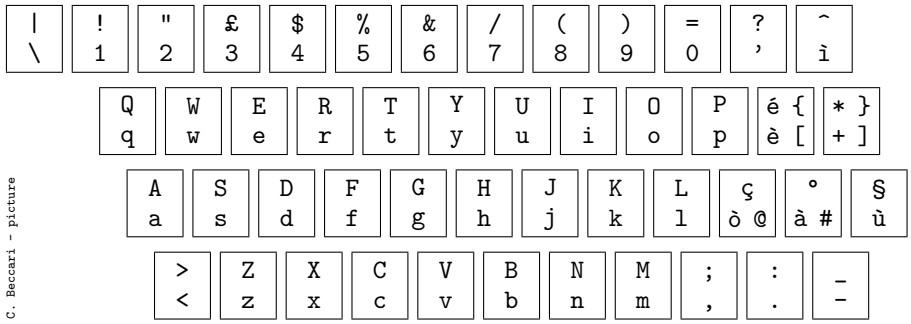


Figura 4.2: Schema della tastiera italiana sulle piattaforme Windows

di gestire (però non può salvare in formati vettoriali – se ne parlerà diffusamente più avanti).

Questi programmi accessori sono utili di per sé e quindi possono essere usati anche indipendentemente dal sistema TEX, tuttavia risultano molto utili per svolgere alla perfezione certe funzioni che solo un programma specializzato sa fare, certamente meglio di quanto potrebbe fare un programma generico dalle 'troppe' funzioni.

Tutti i programmi menzionati, tranne Adobe Acrobat, sono disponibili gratuitamente e sono scaricabili dalla Rete adatti per essere usati sulle specifiche piattaforme di lavoro a disposizione.

4.6 Le tastiere

Questo è un argomento dolente per coloro che hanno la tastiera italiana. Nella tastiera italiana, figura 4.2, sono evidenziati i tasti normali e in particolare sono messe in evidenza le parentesi graffe che hanno una importanza enorme nella composizione dei file sorgente `.tex`.

Si nota però l'assenza dei segni ‘ (accento grave, o backtick, o virgoletta semplice aperta) e ~ (tilde, o legatura, o spazio insecabile). Per introdurre questi segni bisogna ricorrere a espedienti diversi a seconda della macchina usata.

Sulle macchine UNIX (Linux e Macintosh) non dovrebbero esserci grossi problemi, nel senso che esistono combinazioni di tasti che consentono di scrivere virtualmente qualunque carattere dell'alfabeto latino e vari segni e simboli usati normalmente negli scritti in italiano e nelle varie lingue occidentali.

Invece sulle macchine Windows esistono problemi seri che possono venire superati come spiegato qui di seguito.

4.6.1 Le tastiere delle macchine Windows

Il problema esiste con le macchine Windows, almeno fino a alla versione Win7 inclusa; la versione Win8 si comporta come le macchine Linux e Mac grazie ad opportune combinazioni di tasti premuti contemporaneamente o in sequenza. Con gli altri sistemi operativi si hanno alcune possibilità:

1. si tiene aperta la finestra dove si è lanciato l'applicativo Character map (o Mappa dei caratteri se il sistema operativo è in italiano) e ogni volta che occorre un segno non presente sulla tastiera lo si cerca in questa finestra, lo si seleziona e lo si copia, poi lo si incolla nel testo presente sullo schermo.

2. Si sfrutta lo *shell editor* per aprire dentro a questo la mappa dei caratteri disponibili con la codifica impostata per lo *shell editor*, si cerca il carattere assente dalla tastiera e lo si clicca; questo generalmente appare subito nel testo presente sullo schermo; ma può succedere, per esempio, che selezionando 'È' nella suddetta mappa dei caratteri disponibili, sullo schermo non appaia 'È', ma `\‘{E}`. Poco male; la lettura sullo schermo ci rimette un poco, ma l'operazione è corretta ed evita di dover andare a cercare il carattere ‘ assente dalla tastiera. Al massimo questa operazione può risultare un fardello per il correttore ortografico, ma ce ne sono che distinguono le sequenze TEX per gli accenti.

3. Disponendo di una tastiera con il tastierino numerico separato (o premendo il tasto `fn` che cambia significato ad un certo numero di tasti 'ordinari' in modo che si comportino come il tastierino laterale) basta premere il tasto `Alt` seguito da codice numerico ASCII[5] dell'accento grave, che per la cronaca è il numero 96 (oppure 096); rilasciando il tasto `Alt` dopo aver introdotto il codice numerico, sullo schermo appare il segno ‘. Si può usare lo stesso metodo per inserire la tilde il cui codice è 126. Per le maiuscole accentate e per gli altri segni si può procedere come nei punti precedenti oppure ci si scrive da parte una volta per tutte l'elenco dei codici delle lettere che si incontrano più sovente e si usa la tecnica qui descritta per una introduzione abbastanza semplice che non necessita dell'uso del mouse, sgradito ad alcuni utenti.

4. Dal sito `http://www.microsoft.com/en-us/download/details.aspx?id=22339` si scarica il programma Microsoft Keyboard Layout Creator, che permette di configurare la tastiera come si preferisce, attivando tutte le combinazioni di tasti che si desiderano per introdurre tutti i segni disponibili con il driver della tastiera. Bisogna ricordarsi di dare un nome al layout personale che si è creato e di sceglierlo come tastiera predefinita. Molti utenti ridefiniscono il significato dei tasti che contengono segni raramente usati in italiano, come per esempio ç o § o ° e sono contentissimi di questa personalizzazione. Le spiegazioni contenute in `http://www.gialloporpora.netsons.org/`

[5] I caratteri ASCII sono esposti nella tabella 26.1.

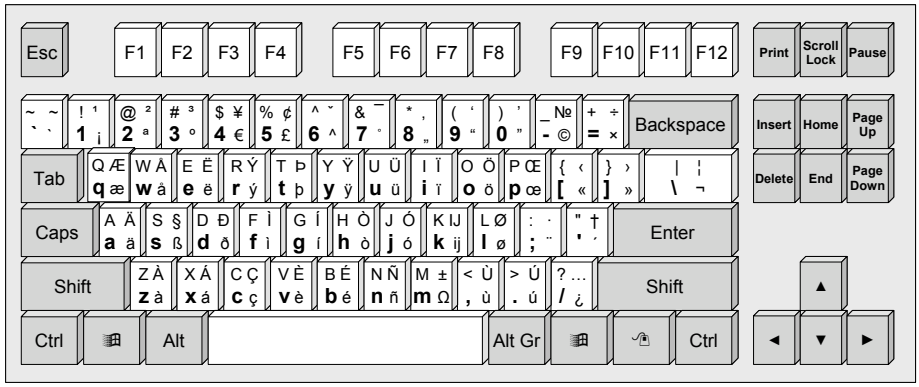


Figura 4.3: Schema della tastiera configurata con il driver EurKey. Tratto dal sito `http://eurkey.steffen.bruentjen.eu/`.

`rimappare-la-tastiera-in-windows-creando-un-file-di-layout/395/` dovrebbero essere particolarmente utili.

5. In un file di macro personali si inseriscono tutte le definizioni, le più mnemoniche possibile, per inserire i segni mancanti dalla tastiera; la definizione di nuovi comandi verrà vista molto più avanti, ma qui si indicano alcune definizioni il cui elenco può essere completato secondo le proprie necessità:

```
\newcommand\A{\‘A}
\newcommand\E{\‘E}
...
\newcommand\U{\‘U}
```

   Come si vede si tratta di definizioni semplicissime, che possono venire ulteriormente arricchite di funzionalità, ma che consentono di evitare la complicazione di battere diversi tasti o di usare il mouse per inserire i segni assenti dalla tastiera italiana.

6. Per le macchine Windows e Linux esiste anche un driver EurKey scaricabile dal sito `http://eurkey.steffen.bruentjen.eu/`[6], che permette di avere un layout di tastiera molto più completo (sebbene anche il driver fornito insieme ai sistemi Linux sia già molto ricco). Il layout viene chiamato “layout europeo” perché dovrebbe essere in grado di introdurre qualunque segno latino con diacritici presente nelle lingue europee. Nella figura 4.3, tratta dal sito suddetto, vi sono mostrati i tasti con l’indicazione dei quattro

[6] Questo tipo di driver esisterebbe anche per le piattaforme Mac; il sito di EurKey rimanda ad un repository `github` da cui è possibile scaricarlo, ma le istruzioni per l’installazione sono, forse, incomplete. D’altra parte i driver per le varie tastiere nazionali delle piattaforme Mac sono così completi che EurKey non sembra una estensione essenziale.

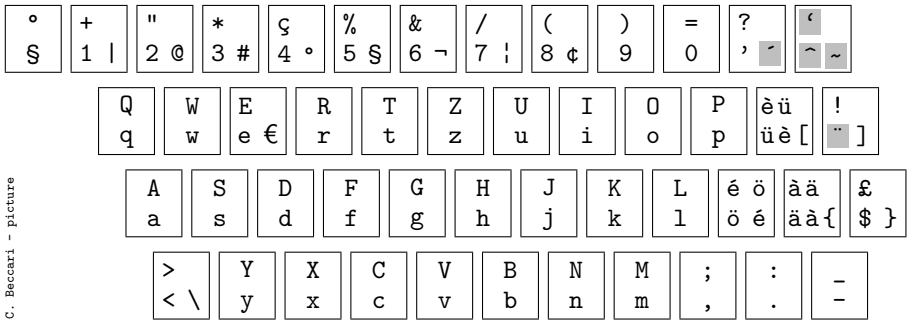


Figura 4.4: Schema della tastiera svizzera sulle piattaforme Windows. Vi si notano gli "accenti morti" evidenziati con lo sfondo grigio, i quali consentono di inserire il loro segno su qualunque lettera che lo accetti. La tastiera è utile per scrivere nelle quattro lingue ufficiali della Confederazione Elvetica, anche se non riporta tutte le vocali accentate dell'italiano e del romancio, che comunque si possono ottenere con gli accenti morti. Le maiuscole accentate si ottengono usando il tasto delle maiuscole permanenti.

segni che si possono comporre con ogni specifico tasto; si arriva a più di 200 segni che coprono quasi completamente le esigenze di tutte le lingue europee.

Se si può segnalare un inconveniente, questo consiste nel fatto che il layout di base è quello USA, ma che i segni aggiuntivi non sono serigrafati né sulla tastiera USA, né su qualunque altra tastiera distribuita commercialmente. O si tiene il disegno della tastiera accanto al PC mentre si digita il codice di entrata, oppure ci si arrangia con piccole etichette da incollare sui tasti con un velo trasparente protettivo (scotch trasparente, per esempio), al fine di evitare la progressiva cancellazione delle indicazioni supplementari via via che si usa la tastiera.

Si richiama il fatto che gli accenti morti servono anche per le lettere maiuscole; esiste persino un tasto, AltGr+m) che permette di passare alla digitazione in greco.

L'uso di questa tastiera e dei suoi numerosissimi segni implica la specificazione dell'opzione *utf8* per il pacchetto *inputenc* nel preambolo di ogni documento LaTeX.

Alternativamente, anche se la cosa può sembrare assurda, potrebbe essere conveniente pensare all'acquisto di una seconda tastiera; esistono diverse tastiere QWERTY nazionali o internazionali che consentono di introdurre tutti e 95 i caratteri ASCII premendo al massimo due tasti; se si è disposti ad abituarsi ad una disposizione dei tasti QWERTZ, le tastiere svizzere (figura 4.4) e tedesche sono complete; così come lo sono altre tastiere dei paesi del nord. Quelle inglesi e

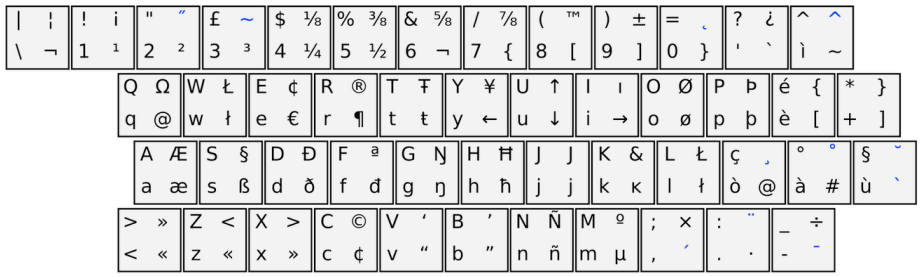


Figura 4.5: Schema della tastiera italiana su Linux senza personalizzazioni. La figura è tratta da Internet, ma il sito non è più raggiungibile.

irlandesi e quelle USA, specialmente se "professional" o "extended", consentono di introdurre anche tutti o quasi tutti i segni previsti per la codifica ISO 8859-1. Spesso queste tastiere possono venire acquistate via Internet da produttori stranieri; spesso sono gli stessi produttori di computer a consentire l'acquisto via Internet e offrono le tastiere più disparate. Ovviamente è necessario che siano dotate del driver necessario per il sistema operativo della propria macchina, altrimenti sarebbero un acquisto inutile, anche se spesso si possono trovare in rete dei driver adatti ad una completa personalizzazione della tastiera.

Per una eccellente presentazione e descrizione delle tastiere nazionali e internazionali si veda il sito `http://en.wikipedia.org/wiki/Keyboard_layout`.

4.6.2 Tastiere Linux

Dopo l'installazione del sistema operativo Linux, in qualunque incarnazione, premere il tasto `AltGr` permette di ottenere diversi altri segni che sono normalmente assenti dalla tastiera italiana.

Dopo l'installazione del sistema la tastiera italiana potrebbe risultare arricchita di nuove funzionalità come appare nella disposizione indicata dalla figura 4.5. I segni marcati in blu sono i tasti morti per i segni diacritici; si noti tra l'altro la presenza sia della cediglia, sia della virgola sottoscritta del rumeno. Vi sono diverse ripetizioni, ma tutto sommato questa disposizione è piuttosto ricca di segni. In ogni caso nella impostazione di sistema che concerne la tastiera è possibile configurare le combinazioni di tasti che si preferiscono per agevolare l'introduzione dei segni che non sono direttamente marcati sui tasti.

L'uso di tutti questi segni implica la specificazione della codifica *utf8* al pacchetto *`inputenc`* nel preambolo di ogni documento LaTeX.

4.6.3 Le tastiere sulle macchine Macintosh

Qui ci riferiamo alle macchine Macintosh moderne, dotate di sistema operativo Mac OS X. Queste macchine, qualunque disposizione di tasti abbiano (italiana, USA, o altra disposizione) dispongono di una comodissima interfaccia grafica

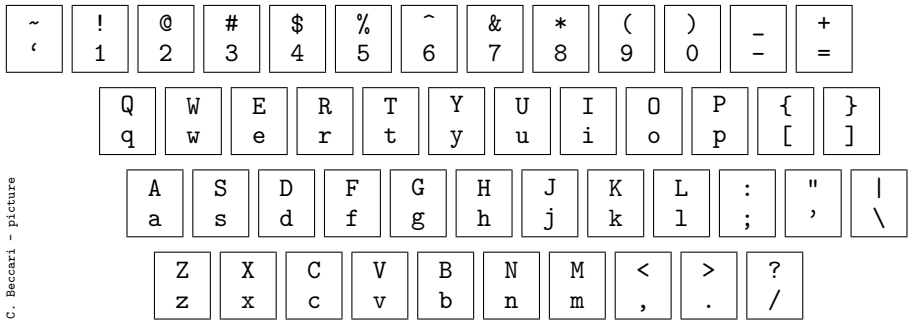


Figura 4.6: Schema della tastiera USA estesa sul MacBook Pro

che permette di avere sullo schermo la disposizione della tastiera e come questa cambia premendo i tasti serigrafati con i segni ⇧ (shift), ctrl, ⌥ (alt), o ⌘ (cmd). Nel disegno della tastiera i tasti morti sono disegnati in colore beige; dopo aver premuto (o cliccato su) un tasto morto la tastiera normale cambia aspetto e presenta solo i caratteri che corrispondono alle lettere ‘accentate’ con il particolare ‘accento’ del tasto morto usato. Dopo un poco si imparano a memoria queste combinazioni di tasti e non è più necessario ricorrere all’immagine della tastiera.

Chi scrive usa abitualmente una tastiera estesa con la disposizione dei tasti secondo lo schema USA, priva, quindi, di qualunque segno accentato, come si vede nella figura 4.6; sono disponibili due driver, quello ‘USA’ e quello ‘USA extended’, il secondo driver essendo il driver più perfomrante. Tuttavia chi scrive ormai esegue automaticamente l’operazione, per esempio, di battere assieme i tasti alt e ‘ e successivamente il tasto E per avere nello schermo il carattere ‘È’.

4.6.4 Le tastiere virtuali

Abbiamo visto sopra che sulle macchine Mac esiste un applicativo che permette di mostrare il layout della tastiera in uso e che permette di scrivere anche i segni che non sono serigrafati sui tasti. Lavorando con la mano sinistra sui tasti di controllo e con la mano destra sul mouse si riesce a scrivere praticamente con ogni tastiera il cui driver sia stato installato nel sistema. Questo è particolarmente utile per scrivere anche in alfabeti diversi da quello al quale corrisponde la tastiera fisica connessa all’elaboratore. Ecco, quindi, come si può agevolmente scrivere in cirillico o in greco. Ma apparentemente questo si può fare solo sulle macchine Macintosh.

Non è così: chi scrive non dispone di una macchina Windows, ma è al corrente che con i sistemi operativi 8 e seguenti anche su queste macchine esista la possibilità di visualizzazione a schermo della tastiera interattiva come sui Mac. Inoltre oggi si stanno diffondendo gli schermi *touch screen*.

Per lo più questa funzionalità *touch screen* viene riservata ad alcuni giochi o a operazioni grafiche. Ma questi elaboratori possono davvero funzionare senza una tastiera materiale, consentendo all'utente di servirsi del disegno della tastiera riportato sullo schermo; l'utente tocca i tasti virtuali sullo schermo e il testo appare nel documento che si sta scrivendo. Estremamente comodo e funzionale, anche se costringe l'utente a scrivere in una posizione innaturale, su una tastiera sostanzialmente verticale, come lo schermo, e non orizzontale come una tastiera materiale.

Tuttavia se si dispone di una tastiera italiana ma si desidera usare una tastiera USA, per esempio, basta averne caricato il driver, e averlo azionato toccando l'apposta icona nella barra inferiore o superiore dello schermo. Se si vuole usare una tastiera greca, lo stesso, basta averne caricato il driver e aver toccato la sua icona per attivare quella tastiera. E così vale per ogni tastiera con uno schema dei tasti diverso da quello della tastiera fisica in uso.

Oggi poi si stanno diffondendo dei leggeri e non tanto piccoli elaboratori portatili dotati di schermo *touch screen*; essi possono anche essere privi di una tastiera fisica. Se dispongono di una tastiera (*tablet* PC), si può ruotare lo schermo in posizione orizzontale, coricandolo sopra la sua tastiera, e usarlo per scrivere con le mani nella posizione solita ed ergonomicamente valida. Questi piccoli elaboratori possono essere usati tranquillamente non solo come tastiere virtuali, ma se sono di potenza adeguata (disco, RAM, velocità, dimensioni dello schermo) possono anche sostituire completamente un PC "normale", sia esso da tavolo o da viaggio.

Il problema tastiera con questi dispositivi cessa di essere un problema.

4.7 Gli shell editor

Nelle versioni precedenti alla 0.90 di questa Introduzione non si era deliberatamente fatto nessun cenno agli *shell editor* perché si riteneva che questo fosse un argomento facoltativo, non direttamente collegato alla composizione tipografica con LaTeX.

In un certo senso questo argomento è ancora valido; in rete e in commercio esistono tanti diversi *shell editor* adatti ai vari sistemi operativi che forse c'è addirittura l'imbarazzo della scelta; tuttavia si è ora ritenuto che fosse necessario dedicare un cenno a qualche *shell editor*, specialmente quelli che sono direttamente disponibili con i software di installazione del sistema TeX.

Di alcuni *shell editor* basta conoscere il nome, cercare su Internet, e scaricare dall'opportuno sito il programma di installazione, provvedere all'installazione e poi farne uso in modo da imparare progressivamente a sfruttarli al meglio.

Essi hanno in comune alcune caratteristiche:

- permettono di creare e correggere (editare, in gergo) file puramente testuali, non formattati, o comunque di salvare il file in modo che contenga solo il testo e il mark-up, senza nessuna informazione sulla formattazione del

testo, perché questa operazione è rimandata alla compilazione sotto l'azione dei file di classe e dei file di estensione usati. Quasi tutti permettono di scegliere fra diverse codifiche dei caratteri immessi da tastiera; i più recenti e meno caratteristici di una particolare piattaforma consentono l'uso della codifica UNICODE. Si parlerà diffusamente più avanti in diversi capitoli della necessità di scegliere opportunamente la codifica per i file sorgente (vedi il capitolo 26).

- Durante il processo di editing è possibile avere in azione uno o più correttori ortografici per una o più lingue. Inoltre alcuni editor possono eseguire operazioni di taglia e incolla non solo per gruppi di righe, ma anche per blocchi rettangolari di un certo numero di righe che si estendono orizzontalmente per un certo numero di colonne; con questi editor il mouse ha anche delle funzioni di selezione più estese di quelle che sono normalmente disponibili.
- Per quel che riguarda il testo da comporre è generalmente possibile scegliere la codifica (encoding) dei caratteri introdotti con la tastiera e, se questa è di tipo avanzato, è anche possibile scegliere la codifica UNICODE, al punto da poter introdurre direttamente caratteri di altri alfabeti, anche retrogradi, e gli ideogrammi delle lingue dell'estremo oriente.
- Se si sceglie di suddividere il file del documento in tanti spezzoni di dimensioni minori, essi generalmente consentono di editare diversi file alla volta.
- Essi generalmente consentono di scrivere e salvare su disco i file sorgente per qualunque programma del sistema TEX; come si vedrà, i programmi di composizione di questo sistema non si limitano al solo LATEX, e non si limitano solamente alla composizione di testi, ma anche alla creazione di disegni vettoriali o a matrici di punti, quali sono, ad esempio, gli stessi caratteri con cui vengono composti i testi.
- Permettono di cliccare su un bottone virtuale con il puntatore del mouse, bottone scelto fra una serie di bottoni per scopi analoghi, oppure di scegliere il programma di composizione da una lista di un menù a caduta, per salvare il file e per lanciare direttamente il programma di composizione.
- Dopo l'esecuzione del programma di composizione, se non ci sono stati errori, lo *shell editor* apre direttamente il programma di visualizzazione adatto al formato di uscita del documento composto. Se invece ci sono errori, lo *shell editor* permette di aprire il file "di registro" nel quale sono elencati gli errori in modo da mettere in relazione le righe diagnosticate come erronee, come appaiono nel file di registro, con le righe presenti nel file sorgente così da poter navigare facilmente avanti e indietro fra il file di registro e il file sorgente.
- Nello stesso tempo se non ci sono errori di mark-up, ma il testo composto presenta dei refusi o delle frasi incomplete, erronee, imprecise, oscure, o

altro, lo *shell editor* permette di navigare avanti ed indietro fra la finestra di visualizzazione del documento composto e la finestra dell'editor contenente il file sorgente, in modo da usare la finestra di visualizzazione come se fossero le bozze del documento così da apportare le debite correzioni o modifiche al file sorgente. In genere questa operazione su alcune piattaforme risulta più difficile o impossibile con il formato PDF, ma è sempre possibile con il formato DVI, anche se poi il documento finale verrà prodotto con il formato PDF. Sui sistemi Mac questo inconveniente non si presenta mai perché il programma di visualizzazione integrato in TEXShop è fatto apposta per gestire agilmente i file in formato PDF, che rappresentano il formato di default di tutti i documenti presenti sulle macchine Mac. Ora però sono disponibili alcuni editor multipiattaforma che consentono di spostarsi agevolmente fra punti corrispondenti della finestra di composizione contenente il file sorgente e della finestra di visualizzazione anche con il formato PDF.

- Alcuni *shell editor* sono multipiattaforma, vuoi perché sono scritti in un linguaggio interpretato, per esempio Java, vuoi perché vengono prodotti simultaneamente nelle versioni adatte alle tre piattaforme principali.

- Alcuni *shell editor* sono freeware o, comunque, gratuiti, altri sono shareware; le distribuzioni commerciali del sistema TEX contengono *shell editor* specifici della particolare versione commerciale e quindi sono, per così dire, compresi nel prezzo.

- Alcuni programmi non sono dei veri e propri *shell editor*, ma consentono di visualizzare direttamente il prodotto della composizione oppure permettono di presentare schermate che assomigliano fortemente al risultato finale, nel senso che i comandi di mark-up (almeno alcuni) sono già interpretati in modo da mostrare sulla finestra dell'editor praticamente solo il testo. Di questi programmi si farà nel seguito un breve cenno separatamente dalle brevi descrizioni relative agli *shell editor* veri e propri.

Il sito `http://en.wikipedia.org/wiki/Comparison_of_TeX_editors` è di grande interesse. Vi si trova una grande tabella dove sono nominati virtualmente tutti gli *shell editor* esistenti. Di ciascuno viene detto se si tratta di software gratuito o se è un programma commerciale; per quali piattaforme esiste il programma; se assomiglia ad un editor WYSIWYM (vedi poco più avanti) o se lavora solo sul file sorgente; in più, ovviamente, vengono elencate tutte le caratteristiche illustrate sopra insieme ad altre caratteristiche meno essenziali. Il confronto è molto istruttivo e permette di scegliere il proprio *shell editor* a ragion veduta. In generale quelli che accompagnano nativamente ogni distribuzione sono abbastanza buoni, talvolta ottimi. Apparentemente, però, quel sito lascerebbe intendere che esista un solo editor multipiattaforma, che, se dotato dell'opportuno plug-in, permette di avere a portata di mano tutte le possibili caratteristiche che si possono chiedere ad uno *shell editor*. Chi scrive consiglia di non prendere

quelle informazioni per oro colato: si tratta di informazioni utilissime, ma talvolta incomplete, quindi il lettore si serva pure di quella tabella, ma con giudizio.

4.7.1 Shell editor multipiattaforma

Come già accennato esistono *shell editor* multipiattaforma: lo stesso programma gira su macchine diverse, con hardware e software diversi. Questo è possibile se i programmi relativi sono scritti in un linguaggio interpretato di cui esista l'interprete per tutte le piattaforme, per esempio, il linguaggio Java con la sua Java Virtual Machine.

Altri *shell editor* sono multipiattaforma nell'altro senso, cioè chi li ha prodotti ha compilato i codici sorgente in modo da ottenere i codici eseguibili validi per tutte e tre le piattaforme principali; a parte questo dettaglio tecnico, essi hanno anche lo stesso aspetto da un punto di vista grafico ed hanno esattamente le stesse prestazioni, gli stessi comandi, le stesse scorciatoie, eccetera.

Alcuni sono multipiattaforma nel senso che chi li ha prodotti ha provveduto a generare i file di installazione che ne permettono l'uso immediato, ma solo per due sistemi operativi, mentre per il terzo sono necessarie procedure particolari per l'installazione; in ogni caso l'aspetto grafico e il funzionamento sono sostanzialmente identici.

emacs è un potentissimo *shell editor* molto noto agli utenti dei sistemi UNIX; associato alla sua estensione `auctex` permette di editare file LaTeX con una serie di facilitazioni ben congegnate; questa estensione consente anche di eseguire la compilazione e la successiva previsione dell'aspetto grafico di parti selezionate del file sorgente. In pratica, selezionando una parte del file sorgente e cliccando su un opportuno bottone (o dando da tastiera uno dei numerosissimi comandi abbreviati ottenuti premendo contemporaneamente uno o più tasti di controllo (Ctrl, Shft, Alt, ...) e un altro tasto (lettera, cifra, ecc.)), si può compilare e vedere nel visore del file di uscita quel brano già composto come lo farebbe latex; questa funzionalità è molto utile specialmente quando si devono comporre formule complicate, tabelle complicate, disegni, eccetera.

emacs consente di usare tutte le possibili codifiche dei font in entrata, anche se con le macchine Windows vecchiotte converrebbe impostare l'encoding *ansinew*.

Questo editor è potentissimo, ma molti utenti lo trovano difficile da usare; esso diventa formidabile quando si conoscono a memoria moltissimi comandi abbreviati, cosicché non sia quasi mai necessario ricorrere al mouse, che i puristi non usano mai.

TeXstudio ha ottime caratteristiche per la gestione dei file sorgente; permette di vedere la struttura del documento, di disporre di menù a discesa per selezionare i vari simboli matematici, i vari ambienti, i caratteri non ASCII, con comodi pulsanti a lato della barra a sinistra del panello dell'editor.

Consente di fare il passaggio dalla finestra del file sorgente alla finestra del file composto e viceversa, purché il commando per lanciare pdflatex contenga l'opzione `-synctex=1`.

La prima volta che si usa TeXstudio, esso fa una scansione del sistema per vedere quali strumenti siano già installati e li inserisce direttamente nella configurazione del programma, configurazione che in un secondo tempo l'utente può cambiare a piacimento, in particolare per quel che riguarda l'encoding dei caratteri in entrata.

Chi scrive l'ha usato e trova che sia molto buono. Esso può essere fatto lavorare anche con il visualizzatore esterno SumatraPDF con il quale è possibile sincronizzare le finestre di composizione e di visualizzazione come con il visualizzatore interno, ma lasciando inalterate tutte le altre funzionalità di SumatraPDF. Si tenga presente che il formato PDF diventerà de facto lo standard per la composizione dei documenti.

Per quel che riguarda il riconoscimento della codifica esso è automatico per la codifica *utf8*, ma quando il file da aprire non è scritto con questa codifica esso se ne accorge e apre una finestra di dialogo perché l'utente scelga la codifica opportuna. Tutto ciò non è gestibile con le righe di autoconfigurazione; si veda la descrizione di TeXShop nella pagina 73.

TeXmaker è il programma originale da cui è nato TeXstudio; hanno in comune un certo look generale delle finestre e molte funzionalità; quello che (per ora) manca a Texmaker è la capacità di interpretare i commenti di autoconfigurazione; sono talmente utili che questa mancanza può essere accettata solo da chi non ne conosce l'utilità; si veda la descrizione di TeXShop nella pagina 73.

TeXworks è un programma freeware multipiattaforma derivato dal programma TeXShop per i calcolatori Mac; è uno *shell editor* la cui versione stabile viene installata direttamente sulle macchine Windows e Mac quando si installa TEX Live o MacTEX rispettivamente; per le macchine Linux bisogna installarlo appositamente come spiegato qui appresso. In ogni caso si possono scaricare dal suo sito le versioni in evoluzione per Windows e Mac; per installare TeXworks sulle macchine Linux bisogna scaricare il pacchetto dei file sorgente e bisogna completare le librerie in dotazione al sistema; poi diventa abbastanza facile seguire le indicazioni nel sito `http://code.google.com/p/texworks/wiki/Building` per la particolare versione del sistema operativo Linux che si sta usando.

Per gli utenti della distribuzione MiKTEX non c'è per ora nessun suggerimento che il programma sia già disponibile (in fondo è ancora — oggi siamo a dicembre 2010 — alla versione 0.3.0); chi ne ha sentito parlare, lo trova già lì, nella cartella `.../MiKTeX2.9/miktex/bin/`; se si vuole, se ne può creare una "scorciatoia" da trasferire sul desktop, in modo da poterne fare l'uso che si desidera. La prima volta che lo si usa esso cerca i file di cui ha bisogno e configura tutto o quasi quello che serve.

TeXworks è in grado di usare l'encoding UNICODE e di lanciare tutti i principali programmi di composizione del sistema TEX. Il suo visualizzatore integrato, però, visualizza direttamente i file di uscita in formato PDF e permette di passare direttamente con un semplice click del mouse dalla finestra di composizione al punto corrispondente della finestra di visualizzazione e viceversa; il visualizzatore è molto rapido. Le dimensioni delle due finestre sono adeguate a riempire bene uno schermo con rapporto di forma 16 : 9, o altri simili schermi panoramici, in modo da averle affiancate senza sovrapposizioni.

Dispone di un comodo correttore ortografico usabile per diverse lingue; solo il dizionario ortografico per l'inglese è preinstallato; per l'italiano bisogna scaricare lo stesso file di correzione ortografica che usa OpenOffice.org e metterlo in una apposita cartella, e così bisogna fare per le altre lingue il cui dizionario ortografico non sia già preinstallato. Con la distribuzione di MiKTEX 2.9 le lingue preinstallate sono aumentate: oltre all'inglese americano ci sono anche i file di correzione ortografica per il francese e per il tedesco; se si vuole aggiungere la possibilità di correggere anche l'italiano, o qualunque altra lingua, bisogna farlo prima di avviare la prima volta TeXworks; bisogna inserire i due file `it_IT.aff` e `it_IT.dic` del correttore ortografico di OpenoOffce.org nella cartella `...\Programmi"92MiKTeX"922.9"92hunspell\dicts\` insieme agli altri file dello stesso genere per le lingue già installate. Quando si apre TeXworks per la prima volta esso si autoconfigura e riconosce la presenza dei file in questione, selezionabili per la verifica ortografica del testo italiano. Al momento TeXworks non è ancora in grado di aggiungere voci al dizionario ortografico, ma è una delle operazioni previste durante lo sviluppo di questo editor.

Consente di interpretare direttamente certe speciali linee di commento all'inizio dei file sorgente, in modo da poter specificare sia il programma di composizione desiderato, sia l'encoding del particolare file, sia, importantissimo, il nome del file principale, il master file, in modo che il progetto possa essere gestito senza file ausiliari; si veda la descrizione di TeXShop nella pagina 73, sebbene TeXworks accetti sintassi di autoconfigurazione leggermente diverse, esso interpreta correttamente anche quelle di TeXShop. Questo *shell editor* dispone anche della prerogativa del completamento dei comandi: si comincia a scrivere un comando, poi si batte il tasto Tab o $\longrightarrow$ finché sullo schermo non appare il comando completo desiderato. Nel file di Aiuto sono elencate anche le numerose scorciatoie che esso consente di usare quando si desidera evitare l'uso del mouse. Certamente in poco tempo verranno messi in luce sia gli errori, sia le prestazioni ancora mancanti e, per quando sarà arrivato alla versione 1.0 (probabilmente anche prima), diventerà de facto l'unico *shell editor* per tutte le piattaforme.

Questi *shell editor* multipiattaforma sono i più consigliabili in generale; se

poi uno desidera avvalersi di prestazioni particolari su una macchina particolare, allora può considerare la possibilità di installare uno *shell editor* specifico fra quelli descritti nelle sezioni seguenti.

4.7.2 Shell editor per le macchine Windows

Per le macchine Windows il disco di installazione TEX Live offre la possibilità di installare il freeware TeXnicCenter; il programma di installazione del sistema MiKTEX, ProTEXt, segnala la possibilità di installare lo shareware WinEdt o il freeware Texmaker indicandone i siti da cui eseguire il download. In rete si trova anche gli editor freeware TeXstudio e LED. Esiste anche una versione del freeware emacs per le macchine Windows.

WinEdt è un programma shareware ottimo, specialmente sviluppato per le macchine Windows del cui sistema operativo sfrutta non poche librerie dinamiche; usa solo la codifica dei font in entrata specifica delle macchine Windows su cui opera, offre al "tastierista" la possibilità di immettere una grande quantità di caratteri non rappresentati direttamente sulla tastiera, segni speciali e lettere con ogni possibile segno diacritico; permette di introdurre quasi tutti i segni compresi nello standard ISO 8859-1 (Latin 1); le versioni più recenti possono anche usare la codifica UNICODE; quindi nel preambolo dei documenti da comporre è necessario specificare l'opzione della codifica da usare, sia essa *latin1* o *utf8*.

Il programma permette di disporre di una moltitudine di menù che consentono con un solo click del mouse di introdurre nel testo sorgente le macro che individuano i simboli e i segni degli operatori matematici di vario genere.

Esso permette di disporre di maschere (*template*) per ogni genere di ambiente testuale, grafico o matematico, cosicché inserita la maschera nel file sorgente non resta che riempirla con le informazioni specifiche del materiale da comporre.

Così come è distribuito, il programma dispone di una moltitudine di vocabolari per la verifica ortografica; per motivi storici la lista di parole che compongono il dizionario per la lingua italiana è molto ricco, ma le parole tronche sono prive di accento; questo apparente inconveniente viene eliminato con l'uso, perché ogni volta che una parola tronca correttamente accentata (e, ovviamente, anche ortograficamente corretta) presente nel file sorgente viene segnalata come errata, basta un click di mouse per aggiungerla al dizionario; dopo pochi giorni d'uso le parole tronche che l'autore usa abitualmente sono tutte inserite nel dizionario; successivamente la mancanza di altre parole tronche viene segnalata solo raramente.

Il programma permette di definire un "progetto", vale a dire di definire in una struttura interna, in un file con estensione `.prj`, l'elenco completo dei file che compongono un dato documento e permettono di registrare

anche le impostazioni per il o i dizionari da usare per quel documento, così come altre impostazioni in merito, per esempio, alla gestione delle righe che compaiono troppo lunghe per restare completamente nello schermo.

TeXnicCenter offre prestazioni simili ma meno estese di quelle che si possono avere con WinEdt; gli utenti lo trovano (oltre che gratuito) più semplice da usare di WinEdt, ma le prestazioni sono, secondo chi scrive, decisamente inferiori. Recentemente sembra che la manutenzione di questo programma sia cessata, tanto che prima veniva installato di default insieme a MiKTEX, mentre ora le installazioni di MiKTEX includono TeXstudio.

LED è un programma freeware con alcune caratteristiche particolari, come per esempio quella di sfruttare bene gli schermi "panoramici" dei moderni PC e laptop; infatti è possibile tenere aperte simultaneamente una accanto all'altra le finestre del testo sorgente e quella del testo composto (solo però in formato DVI), cosicché la navigazione da una finestra all'altra avviene in modo semplicissimo e comodissimo per controllare errori, refusi, imprecisioni o altro. La schermata del testo composto viene aggiornata automaticamente ogni volta che si clicca sul bottone per l'esecuzione della composizione in formato DVI, che, data la velocità dei PC moderni, dà quasi l'illusione di una composizione sincrona.

Inoltre le funzionalità di LED sono simili a quelle degli altri programmi già descritti; purtroppo non dispone di un visore diretto di file in formato PDF, per cui bisogna ricorrere ad un programma esterno, per esempio Adobe Reader, che però non si integra perfettamente con il programma LED, non altrettanto bene quanto il suo visore interno per i file in formato DVI, cosicché si perdono un poco le prestazioni che LED offre con quest'ultimo formato.

TextPad è uno shareware leggermente più economico di WinEdt (32 € contro 40 €) e anche leggermente meno performante; molti sviluppatori di software lo usano perché permette di gestire file anche di grandi dimensioni (contrariamente al piccolo Notepad di serie su ogni macchina Windows) ma TextPad garantisce una piccola "invasività" della memoria RAM. Ne esiste anche una versione localizzata in italiano, cosicché, almeno per lo *shell editor*, gli utenti poco ferrati in inglese si sentono a loro agio.

Kile è uno *shell editor* che opera solo su macchine Linux e che abbiano l'interfaccia grafica KDE (vedi poco più avanti); per le macchine Windows esiste il programma CygWin che permette di emulare il sistema operativo UNIX/Linux all'interno di un apposito ambiente grafico; dentro questo ambiente CygWin è possibile installare una versione dell'interfaccia grafica KDE, per cui anche sulle macchine Windows si può usare Kile. Forse sembra complicato tutto questo marchingegno per simulare un ambiente Linux; tuttavia esistono moltissimi altri programmi freeware che possono venire installati all'interno di CygWin, oltre che sulle macchine Linux, e che

consentono di eseguire processi di elaborazione mai previsti o non ancora previsti per le macchine Windows.

TeXstudio È uno *shell editor* multipiattaforma ed è già stato descritto nel paragrafo 4.7.1. Va notato che ultimamente viene installato insieme a MiKTEX che, invece, precedentemente installava TeXnicCenter.

Texmaker È uno *shell editor* multipiattaforma ed è stato già descritto nel paragrafo 4.7.1; alcuni lo preferiscono a TeXstudio, forse perché non hanno ancora scoperto l'utilità dei commenti di autoconfigurazione, che invece TeXstudio riconosce.

4.7.3 Shell editor per le macchine Linux

Con le macchine Linux si ha una grande disponibilità di programmi adatti a scrivere ed editare file testuali, anche perché in questo sistema si fa un grande uso di *scripting*, cioè di programmi scritti senza particolari formattazioni, anzi, spesso contenenti solo i caratteri da 32 a 127 della pagina ASCII dei caratteri[7].

Gli utenti fortemente coinvolti con una macchina Linux e che usano lo scripting in modo esteso, generalmente si servono del programma vim oppure del programma emacs; Per usare un vero *shell editor* appositamente previsto per l'uso con LATEX, si può ricorrere al programma Kile, purché l'interfaccia grafica sia l'interfaccia KDE; attenzione, però; su una vera piattaforma Linux, Kile necessita della presenza di un installazione TEX Live conforme alle specifiche del consorzio Debian.

vim è un programma con una interfaccia grafica virtualmente assente; molto veloce e molto efficace nelle mani di un utente di Linux molto esperto, ma poco utile ad un utente "normale". È solamente un efficacissimo "text editor", ma non lo si può chiamare "shell editor" nel senso stretto della parola. Esiste una versione con interfaccia grafica che è più facile da usare, ma resta sempre solo un editor di testi.

emacs è già stato descritto come programma multipiattaforma e non sarebbe necessario aggiungere altro. Merita invece segnalare che emacs è installato di default in ogni distribuzione delle varie implementazioni del sistema operativo Linux; quindi ogni utente Linux ne dispone per il solo fatto di usare una macchina con sistema operativo Linux.

Conviene ricordare di usare sempre l'opzione `auctex` che fa diventare emacs un vero *shell editor* per LATEX, oltre a consentire la visualizzazione quasi sincrona di tratti selezionati del file sorgente.

[7] Sarebbe la prima mezza pagina di ogni codifica di caratteri che contenga solo 256 caratteri, oppure la prima (mezza) pagina delle ulteriori numerose pagine ognuna di 256 caratteri, come la codifica UNICODE. La codifica ANSI contiene una unica pagina di 256 caratteri; le varie codifiche ISO 8859 contengono ciascuna una sola pagina di 256 caratteri. Per tutte le codifiche, però, la prima mezza pagina della prima o dell'unica pagina contiene i caratteri ASCII.

Xemacs è una versione di emacs più avanzata e che già contiene alcune delle prerogative del plug-in auctex, il quale è ugualmente adatto anche a Xemacs. emacs è normalmente installato insieme al sistema Linux; Xemacs invece va installato deliberatamente. Esiste una versione di Xemacs anche per le macchine Windows.

Kile per i sistemi Linux che dispongono del "K Desktop Environment", in sigla KDE, è disponibile anche lo *shell editor* Kile: KDE Integrated LaTeX Environment; è dipendente dalla presenza di una installazione TeX Live, certamente quella Debian compatibile sulle macchine con una versione Debian di Linux; non è escluso che aualsiasi instalalzione TeX Live possa essere sufficiente se si è fatto "credere" al sistema operativo che sia già installata la versione Debian di TeX Live. Si tratta di un ottimo *shell editor* con tutte le funzionalità necessarie e già descritte per altri *shell editor*; in più esso ha indicazioni grafiche che permettono di controllare gli ambienti testuali o matematici, in modo che si possa essere sicuri che ogni apertura di ambiente abbia la corrispondente chiusura (questo è il minimo che si può richiedere e, in un modo o nell'altro, quasi tutti gli *shell editor* offrono questa funzionalità) ma permette anche di "chiudere" ciascun ambiente: vale a dire si può contrarre sullo schermo tutto l'ambiente ad una sola riga, in modo che lavorando sul file sorgente si possa esaminare con più semplicità il testo che precede e quello che segue l'ambiente; cliccando su una piccola icona a forma di ⊞, l'ambiente viene nuovamente espanso; si tratta di una funzionalità grafica molto utile.

Kile permette anche di sfruttare il completamento dei nomi; vale a dire che quando si comincia a scrivere una macroistruzione, per esempio, il programma dopo qualche carattere presenta un menù a discesa con tutti i nomi che cominciano con quelle lettere e fra i quali si può scegliere il nome giusto. Per gli ambienti non appena si scrive lo statement di apertura, Kile inserisce subito dopo lo statement di chiusura; se l'ambiente richiede obbligatoriamente dei parametri, esso inserisce anche le parentesi graffe che li racchiuderanno; sta poi al "tastierista" di completare il gruppo dei parametri e di inserire il contenuto dell'ambiente. Ad alcuni questa funzionalità piace moltissimo, e in generale è considerata una funzionalità molto utile per qualsiasi editor; altri ne sono un poco infastiditi; ovviamente si tratta di gusti personali, per venire incontro ai quali Kile permette di configurare la finestra di editing con o senza questa funzionalità.

Quando si esegue una composizione, diciamo con pdflatex, Kile provvede direttamente a ripetere la compilazione quante volte occorre affinché tutti i riferimenti incrociati siano corretti, che la bibliografia contenga tutti i riferimenti correttamente collegati con i richiami, che l'indice analitico sia aggiornato, eccetera. Anche questa funzionalità può venire disabilitata per la lavorazione iniziale di un documento, e riattivata nella fase finale, cosicché si possa essere sicuri che il documento sia globalmente completo e corretto.

Le macchine Linux che operano con l'interfaccia grafica Gnome possono comunque attivare una parte ridotta dell'interfaccia KDE in modo da poter usare ugualmente Kile, oppure possono ricorrere al programma seguente.

gedit-LaTeX-plugin gedit è il semplice editor testuale con interfaccia grafica disponibile sulle macchine Linux con il desktop gnome. Questo semplice editor accetta l'innesto di plug-in, e il gedit-LaTeX-plugin è quello che permette di gestire i file LaTeX attraverso la sua semplice interfaccia grafica. Ma questo plug-in offre numerose prestazioni utili per la gestione, la modifica e l'esecuzione/compilazione dei file LaTeX. Precisamente consente di completare i comandi; controlla la sintassi del mark-up di LaTeX; permette di avere come Kile, Texmaker e TeXstudio una finestra laterale con la struttura di un documento complesso e permette di navigare facilmente da un punto all'altro dei file sorgente; dispone di un certo numero di "Wizard" che svolgono diversi compiti; gestisce anche i file sorgente da elaborare con BibTeX; consente di usare e, volendo, di modificare gli schemi iniziali dei documenti da comporre; consente di creare degli "snippets" per estendere la gestione e il controllo dei comandi e degli ambienti personali allo stesso modo di come esso fa da solo per i comandi e gli ambienti standard di LaTeX e dei pacchetti più comuni; permette di creare degli strumenti di compilazione personalizzati; dispone di un visualizzatore PDF integrato, come TeXworks e TeXShop, e grazie a questo fatto ha lo stesso tipo di sincronizzazione fra il file sorgente e in file composti in formato PDF. Dispone degli strumenti necessari per il controllo ortografico.

La pagina iniziale di questo gedit-LaTeX-plugin creato da Michael Zeising, si trova in `http://www.michaels-website.de/?page_id=6`, ma il plug-in, che consiste sostanzialmente di un programma scritto in Python, si trova nel sito `http://sourceforge.net/projects/gedit-latex/files/`. Ovviamente questo plug-in dipende da altri pacchetti e altre librerie generalmente presenti su ogni macchina Linux; all'occorrenza non è difficile installare le librerie mancanti.

Apparentemente gedit e quindi questo plug-in sono installabili anche sulle macchine Macintosh con sistema operativo Mac OS X.

TeXworks è lo stesso programma descritto per le macchine Windows, solo che in rete si trovano alcune distribuzioni binarie già usabili e installabili con i soliti strumenti di installazione delle specifiche distribuzioni Linux; altrimenti per tutti è disponibile la "tarball" con i file sorgente ed è possibile compilare il programma con le caratteristiche che si desiderano, ma più propriamente con le funzionalità e le librerie presenti sulla propria macchina.

TeXstudio è lo stesso programma descritto per Windows ed ha le stesse prestazioni oltre alle proprie specifiche, che gli consentono di fare molte più cose e meglio. È importante ricordare che TeXstudio è in grado di intepretare i commenti di autoconfigurazione di TeXShop, cosa che Texmaker (ancora) non è in grado di fare.

Texmaker è lo stesso programma descritto per le macchine Windows, ma sul suo sito c'è anche un certo numero di eseguibili già predisposti per diverse distribuzioni Linux. Anche con questo programma esiste la possibilità di costruirsi il programma personalizzato per la propria macchina.

Si veda la descrizione svolta nella sezione relativa alle macchine Windows per conoscerne le funzionalità. Molto comodo per usare l'encoding UNICODE, ma questo lo fa anche Kile. I due programmi hanno caratteristiche simili.

4.7.4 Shell editor per le macchine Macintosh

Qui ci si riferisce a macchine Mac recenti, operanti con il sistema operativo Mac OS X; questo è un sistema UNIX con l'interfaccia grafica raffinata tipica delle macchine Mac. Su queste macchine possono venire montati anche i programmi validi per Linux e appositamente integrati con l'interfaccia grafica per Mac. Quindi buona parte di ciò che si è detto per le macchine Linux è applicabile anche alle macchine Mac; tuttavia esistono programmi specifici creati e sviluppati per queste macchine; sarebbe opportuno farne uso proprio per sfruttare al meglio le specificità di questo sistema operativo, per il quale ogni documento è generalmente scritto solo usando i formati PDF, oppure RTF, oppure XML (be' molte parti pre-esistenti scritte in HTML sono ugualmente usate, ma generalmente non riguardano i documenti creati dagli utenti).

TEXShop è lo *shell editor* che accompagna di default il sistema TEX, quando questo viene installato con MacTEX. Dal punto di vista dell'editing del testo questo editor non è fortissimo, ma offre tutte le funzionalità principali descritte per gli altri *shell editor*; forse manca di piccole funzionalità di secondaria importanza; esso, dalla versione 2 in avanti è stato molto arricchito di nuove funzionalità e può reggere benissimo il confronto con gli altri *shell editor*.

Invece possiede nativamente una funzionalità unica che normalmente manca negli altri *shell editor*, vale a dire la possibilità di navigare nei due sensi fra la finestra di editing testuale e quella di visualizzazione del testo composto. Questa funzionalità è formidabile tutte le volte che si usa il sistema TEX, ma è particolarmente gradita sulle macchine Mac, visto che il formato di default dei documenti composti è comunque il formato PDF. Certo è possibile configurare il motore di composizione affinché il formato di uscita sia quello DVI, ma la funzione di visualizzazione attraverso il programma integrato in TEXShop richiede comunque sempre il formato PDF, per cui l'eventuale file DVI prodotto viene subito "distillato" nel formato PDF. Si può installare un visualizzatore come xdvi per il formato DVI e si può configurare TEXShop in modo che usi quel visualizzatore, invece del suo visualizzatore interno, ma questa sembrerebbe la strada per rendere complicate le cose semplici.

TeXShop è stato il primo *shell editor* a servirsi dei commenti di autoconfigurazione, conosciuti anche con il nome di *righe magiche*. Questi consistono in normali righe di commento all'inizio dell'unico o del primo file sorgente del documento da comporre (in realtà alcune righe magiche sono utili anche nei file secondari) che lo *shell editor* sa interpretare in modo da cambiare temporaneamente la propria configurazione oppure per eseguire azioni particolari; le righe magiche sono le seguenti:

`% !TEX encoding =` ⟨*codifica*⟩
`% !TEX TS-program =` ⟨*programma di composizione*⟩
`% !TEX root =` ⟨*main file completo di estensione*⟩
`% !TeX spellcheck =` ⟨*lingua_varietà*⟩
`% !BIB TS-program =` ⟨*programma di estrazione bibliografica*⟩

dove:

- la ⟨*codifica*⟩ è uno dei nomi delle codifiche usabili dallo *shell editor*, per esempio `Latin1`, `UTF-8 Unicode`, eccetera;
- ⟨*programma di composizione*⟩ può essere uno dei programmi principali per comporre con LaTeX, `pdflatex`, `XeLaTeX`, `LuaLaTeX`, eccetera, ma può essere anche un qualsiasi altro programma di composizione del sistema TeX;
- ⟨*main file completo di estensione*⟩ è il nome del file principale per la composizione di un dato documento; se il documento è composto con un solo file sorgente, questa riga magica è inutile, ma non crea problemi se la si usa; essa invece è importante per i vari file inclusi dal main file, perché quando si vuole compilare e si clicca l'apposito bottone dello *shell editor* quando esso è aperto su un file secondario, la compilazione parta con il main file e non con il file secondario il quale, in quanto file incluso, non contiene il preambolo e non può quindi venire compilato autonomamente;
- ⟨*lingua_varietà*⟩ contiene la sigla della lingua seguito dalla sigla della varietà; per esempio il suo valore per l'italiano è semplicemente `it_IT`, mentre per l'inglese potrebbe essere `en_US` o `en_UK` a seconda che per il controllo ortografico si voglia usare il dizionario dell'inglese americano o dell'inglese britannico;
- ⟨*programma di estrazione bibliografica*⟩ può valere `Bibtex` oppure `Biber`.

Si noti che specificare il dizionario ortografico con TeXShop non è importante e forse è dannoso, in quanto questo *shell editor* riconosce da solo quale dizionario usare anche in un documento multilingue; per gli altri *shell editor* in grado di interpretare queste righe magiche, invece, è utile specificare il correttore ortografico perché non sono in grado di riconoscere da soli la lingua in uso.

Si noti anche che il ⟨*programma di estrazione bibliografica*⟩ preconfigurato per ogni *shell editor* può essere uno solo per ogni documento, anche se lo *shell editor* può essere preconfigurato per usare entrambi; ma quando si clicca il bottone o si sceglie la voce di menù 'Bibliography' viene usato quello dei due programmai specificato nella riga di autoconfigurazione.

Un'ultima osservazione in merito alle righe magiche: la codifica del main file e dei file inclusi deve essere la stessa e deve corrispondere all'opzione specificata al pacchetto `inputenc`. Non è attraverso i commenti di auto-configurazione che si può cambiare codifica ad un file sorgente; si può solo adattare lo *shell editor* alla codifica del file sorgente.

Aquamacs è una versione speciale di emacs adattata all'interfaccia grafica dei sistemi Mac; dalla versione 2.0 in poi è particolarmente indicata per il sistema operativo Mac OS X 10.5 e per le versioni successive.

Pur conservando l'intera funzionalità di emacs, contiene una vasta serie di menù con i comandi che normalmente si trovano nei programmi per le piattaforme Mac; inoltre integra già l'estensione `auctex`, per cui quando si crea o si apre un file LaTeX il programma cambia automaticamente il suo aspetto e la sua barra dei comandi per mostrare all'utente quanto è disponibile nel caso si debba editare un file LaTeX. Funziona con lo stesso correttore ortografico di TeXShop. Se il visualizzatore PDF che si associa a Aquamacs è Skim, esso consente la sincronizzazione fra i file sorgente e il file composto in formato PDF più o meno con la stessa funzionalità di TeXworks.

Un suo vantaggio molto utile nella gestione del file sorgente è costituito dalla serie di "interpretazioni" dei comandi LaTeX, che consentono di dare un colore diverso, una forma diversa, una grandezza diversa alle parti di testo marcate con il mark-up di LaTeX; un comando `\section` avrà quindi il suo argomento scritto in corpo decisamente maggiore del testo normale, colorato in blu scuro e sostanzialmente in grassetto. Un testo da evidenziare passandolo come argomento al comando `\emph` apparirà composto nella finestra di editing con un font inclinato e di colore contrastante. Lo stesso succede per le note e per gli argomenti di diversi altri comandi. Questi giochi di colore risultano molto utili durante la fase di creazione e di modifica del file sorgente; va da sé che sono attuati tutti gli altri espedienti utili per evitare gli errori più comuni, per esempio mediante l'accoppiamento delle parentesi dello stesso tipo allo stesso livello.

Esso presenta la possibilità di vedere direttamente nella finestra del file sorgente le equazioni e i disegni già trasformati in immagine composta con pdflatex; è ovviamente possibile tornare al file sorgente anche per correggere questi casi, ma se il risultato è soddisfacente, si può proseguire a elaborare il testo sorgente, e il contenuto della finestra del file sorgente scorre sullo schermo portandosi dietro le parti che hanno già subito questa visualizzazione parziale.

La sincronizzazione fra il file sorgente e il file PDF viene eseguita attraverso un programma di visualizzazione dei file PDF che si chiama Skim; questo è un programma indipendente da Aquamacs e va installato per suo conto; esso può venire configurato per agire in sincronia con diversi editor, ma oggigiorno la maggior parte degli editor moderni dispone di un proprio visualizzatore interno già sincronizzato.

emacs viene distribuito anche per operare nativamente con l'interfaccia grafica del sistema Mac OS X, quindi si può usare la combinazione emacs e auctex per operare come sulle macchine Linux; emacs è molto più potente, come editor testuale, di TEXShop, quindi potrebbe essere una buona scelta; tuttavia, anche con l'aggiunta di auctex, lo *shell editor* che si viene a formare può navigare avanti e indietro solo con i file in formato DVI, e questo porta una pesante ipoteca alla comodità di lavorazione del documento.

Kile Poiché il sistema operativo Mac OS X prevede anche l'interfaccia grafica X Windows (tipica dei sistemi Linux) e consente l'installazione dell'interfaccia KDE, si può usare anche lo *shell editor* Kile; ma sotto certi aspetti può sembrare un regresso rispetto a TEXShop.

TextWrangler e diversi altri editor testuali specifici per il sistema Mac OS X possono essere utili, ma non sono *shell editor*; quindi non sono così comodi per la lavorazione di un documento. TeXwrangler, però, può essere molto utile per aprire e leggere i file `.log` perché nativamente questa estensione è collegata ad altri applicativi che chi scrive non trova così comodi. Solo TEXShop, a sua conoscenza, ha una voce di menù che consente di aprire il file `.log` di un progetto.

TeXworks regge il confronto con TEXShop, sebbene sia ancora un po' giovane. Per aggiornarlo rispetto alla versione impacchettata nell'installatore MacTeX, lo si scarica dal suo sito come file `.zip` che però contiene direttamente l'applicazione `TeXworks.app`; basta estrarla dal file `.zip` e metterla nella propria cartella `Applications`, per averlo direttamente funzionante. Ci si ricordi che per impostare la configurazione desiderata, dopo averlo lanciato almeno una volta, anche senza aprire nessun file, bisogna fare come al solito con i programmi Mac, cioè aprire il menù a discesa cliccando sul nome del programma presente nella barra superiore e lì scegliere "Preferences". Nei programmi per Windows e per Linux, invece la scelta delle preferenze personali si trova come ultimo elemento del menù a discesa della voce "Edit" ("Modifica" quando il programma è localizzato in italiano).

Benché questo programma nasca da TEXShop, presenta alcune funzionalità assenti in TEXShop; altre funzionalità sono eseguite in modo diverso, come per esempio la creazione del legame di subordinazione di un file sorgente al suo master file. Così la configurazione dei programmi, delle scorciatoie, dell'encoding, eccetera. È comodo che il pulsante per eseguire la composi-

zione sia presente tanto nella finestra del file sorgente quanto nella finestra del file composto.

Più avanti si vedrà come usare TeXworks per convertire la codifica dei file; anche altri editor consentono di farlo, (aprire un file codificato in un modo e salvarlo codificato in un altro) ma come lo si fa con TeXworks è particolarmente comodo.

TeXstudio ha gli stessi pregi già descritti come *shell editor* multipiattaforma; dispone del suo visualizzatore interno per file PDF già sincronizzato con il file sorgente. Esso, dalla versione 2.4, è anche in grado di interpretare le righe di autoconfigurazione alla TEXShop, cosa che risulta molto comoda in assoluto, ma anche per agevolare il lavoro cooperativo fra autori che usano macchine e sistemi operativi diversi.

Texmaker presenta le stesse funzionalità descritte per i sistemi Windows, e le stesse limitazioni. Nonostante ciò è un ottimo shell-editor; esso è adatto anche alle macchine Mac e contiene un suo visualizzatore interno predisposto per la sincronizzazione.

4.8 Editor quasi WYSIWYG

Sembra opportuno segnalare che esistono dei programmi gratuiti o commerciali che permettono di lavorare i file sorgente `.tex` in maniera quasi WYSIWYG; se ne sconsiglia l'uso perché al fine di presentare la composizione del documento più o meno con la qualità di LaTeX devono rinunciare a molte funzionalità.

Il programma TeXmacs mostra sullo schermo il risultato della composizione proprio come lo farebbe LaTeX, usando persino gli stessi font di default di LaTeX. Non permette di cambiare font né di usare qualunque pacchetto di estensione; il programma è gratuito, ma appunto è estremamente limitato.

Il programma LyX non presenta nella schermata esattamente quello che verrà prodotto con LaTeX, ma ne mostra un buona approssimazione; consente di usare i pacchetti di estensione, quindi in teoria permette di usare LaTeX in tutte le sue sfaccettature. In realtà non è così; spesso consente di caricare pacchetti di estensione, ma per "disegnare" la schermata con l'approssimazione del risultato della composizione con LaTeX, ha bisogno di appositi file di interfaccia che "traducano" il mark up LaTeX nel linguaggio interno di LyX. Il sistema TeX dispone di quasi tremila pacchetti, formati spessissimo da molti file; generalmente chi crea questi pacchetti non usa LyX e quindi non produce i file di interfaccia; come conseguenza usare LyX serve a poco o niente, anzi complica le cose per inserire pacchetti e comandi "nascosti" che disturberebbero la schermata dalle pretese di essere praticamente il risultato finale della composizione con LaTeX. Non è così, ma il suo canto di sirena attrae molti utenti che poi si accorgono troppo tardi che non ne valeva la pena.

Il programma commerciale TeXtures per le macchine Macintosh forse è ancora disponibile, ma sembra che sia successo un disastro nell'ottobre del 2012, quando il

server della società Blue Sky Research è "defunto"; se la produzione di TeXtures fosse cessata, un gran numero di affezionati clienti Mac lo rimpiangerebbe. TeXtures mostrava sullo schermo del calcolatore la finestra di editing, dove si scriveva il testo e i comandi di mark up; nella finestra accanto mostrava il risultato della composizione che esso eseguiva praticamente in tempo reale mentre l'utente scriveva nella finestra di editing. La macchina lavorava molto in fretta, ma era tutta destinata ai due processi, quello di editing e quello di composizione; tutto sommato non era male e l'utente esperto poteva rendere più veloce il suo lavoro se disattivava la finestra di composizione, riattivandola di tanto in tanto per controllare il suo lavoro. Sostanzialmente non è molto diverso da quanto si fa oggi con i moderni editor corredati del visualizzatore PDF interno sincronizzato con la finestra di editing.

4.9 LaTeX e pdfLaTeX

È bene che il lettore sappia che quando si dice LaTeX si intende il linguaggio di mark-up specifico per descrivere le componenti di un documento e di cui parleremo approfonditamente nel resto di questo testo. Nel linguaggio corrente si usa lo stesso nome per indicare il programma che interpreta il mark-up ed esegue la composizione del documento. Quindi esso indicherà non solo il programma latex vero e proprio, ma anche il programma pdflatex. Quest'ultimo ha la caratteristica che il suo formato di uscita è direttamente il formato PDF e non sono necessarie conversioni di nessun genere come è stato più volte menzionato nei paragrafi precedenti.

Vale la pena sottolineare immediatamente le differenze nel file finale che si ottiene direttamente con pdflatex rispetto a quello che si può ottenere mediante le trasformazioni già menzionate, cioè eseguendo in sequenza i programmi latex, dvips e ps2pdf, oppure latex e dvipdfm.

Il programma diretto pdflatex produce un file PDF conforme agli standard e dotato di tutte le caratteristiche che gli standard consentono, dalle annotazioni ai *thumbnail*[8], dai link ai file o siti esterni ai link di navigazione interna del documento. La composizione diretta in formato PDF consente anche una qualità tipografica decisamente superiore grazie alla possibilità di usare la microtipografia; se ne farà una descrizione abbastanza ampia in un successivo capitolo.

Passare attraverso il formato DVI, quello di default che si produce con il programma latex, comunque lo si trasformi, non consente di ottenere tutte le funzioni del formato PDF 'vero', perché il formato DVI, di sua natura, non può contenere certe informazioni. Esso è un modo per descrivere il layout di una

[8] La parola inglese *thumbnail* alla lettera significa *unghia di un pollice*; in italiano si userebbe la parola *unghia*, ma si riferisce a quegli intagli semicircolari sul margine esterno delle pagine che si trovano in grossi volumi di consultazione, come dizionari ed enciclopedie, e che servono per agevolare la ricerca di una particolare sezione alfabetica del volume. In informatica si attribuisce il nome di *thumbnail* alle immagini ridotte delle pagine o delle figure che compaiono spesso sui bordi dei programmi di visualizzazione dei file PDF o nei programmi di sistema che consentono di cercare i file nelle cartelle dei dischi.

pagina tipografica, non di descrivere gli oggetti che ne fanno parte. È vero, esiste la possibilità di aggirare queste limitazioni mediante un comando speciale che, appunto, si chiama `\special`, e che permette di inserire nel file di uscita dei brani di un altro linguaggio che verrà poi interpretato (o ignorato, se non è conforme) dal programma di 'traduzione' del formato DVI in un altro formato oppure per essere visualizzato sullo schermo o per essere stampato. Va comunque precisato che il formato DVI non consente l'uso della microtipografia, e quindi, benché la sua qualità tipografica sia molto buona, non è altrettanto buona di quella che si può ottenere con la generazione diretta del file PDF attraverso il programma pdflatex.

Il formato PDF si chiama appunto Portable Document Format, proprio perché consente di 'portare' i documenti composti su qualunque piattaforma dotata di qualunque sistema operativo, basta che disponga di un visualizzatore di file PDF; e perciò è particolarmente indicato per i *libri elettronici* (o *e-book* come spesso vengono chiamati oggi) previsti non per essere stampati ma per essere 'sfogliati' sullo schermo. Ovviamente il formato PDF va benissimo anche per la stampa e, una volta stampato, le possibilità di navigazione interna o esterna vengono comunque perse.

Fino a poco tempo fa, e in certi ambienti ancora oggi, veniva data una grossa preferenza al formato PostScript; chi lavora in ambiente UNIX e Linux continua a privilegiare il formato PS grazie al fatto che questo formato negli anni '80 e nei primi anni '90 era l'unico che potesse ricorrere a stampanti per la stampa diretta del formato PostScript. Oggi i libri elettronici sono relativamente diffusi e stanno diffondendosi ancor di più con la messa in commercio di piccoli apparecchi capaci di funzionare non solo come telefoni cellulari, ma anche come navigatori di rete, lettori di libri elettronici, riproduttori di file audio, eccetera. Dal punto di vista archivistico oggi esiste una norma ISO che impone l'uso del formato PDF al fine di produrre documenti elettronici archiviabili a lungo termine.

Nel seguito, quindi, ci si riferirà quasi sempre al formato PDF prodotto direttamente con il programma pdflatex.

È bene precisare che i comandi da dare dalla linea di comando, latex o pdflatex, seguiti dal nome del file da elaborare, si comportano come programmi, ma in realtà quei nomi sono solo utili per specificare sia l'interprete dei comandi (il vero motore di composizione) sia il tipo di mark-up contenuto nei file di formato.

Il file di formato è un file che contiene una collezione di definizioni di macroistruzioni che vengono lette come prima cosa dal vero motore di composizione; siccome queste macroistruzioni vengono lette sempre, allora il sistema TeX, fin dall'inizio, ha sviluppato la tecnica di raccogliere le definizioni comuni legate al tipo di mark-up in file speciali, per LaTeX nel file `latex.ltx`, per `plain` TeX nel file `plain.tex`, eccetera; questa tecnica consente di leggere una volta per tutte questi file contenenti le macro comuni mediante una versione 'vergine' dei programmi tex oppure pdftex che, dopo avere letto i file e averli elaborati per trasformarli in linguaggio macchina, li memorizzano sul disco di sistema in file binari con estensione `.fmt`. Quando si lancia latex, in realtà si lancia un programmino di interfaccia, che a sua volta lancia pdftex con l'uscita in formato

DVI, oppure pdftex con l'uscita in PDF, dicendogli anche di usare il file di formato `latex.fmt`.

Questa delucidazione può sembrare capziosa, specialmente se si aggiunge che dal 2005 ogni distribuzione del sistema TEX contiene anche i programmi di composizione etex e pdfetex (vedi più avanti il paragrafo 4.10.2); questi programmi estesi sono compatibili all'indietro con i programmi precedenti, per cui li sostituiscono completamente dal punto di vista funzionale.

Per complicare le cose al neofita, che oltretutto, almeno all'inizio, difficilmente si accorgerà delle differenze, bisogna aggiungere che il programma pdfetex, specificando un opportuno parametro, si comporta in modo esattamente uguale a etex. Dal 2007 pdftex è diventato il programma di composizione di default; l'utente non se ne accorge, ma per chi si dedica alla manutenzione dei programmi la cosa rappresenta una enorme semplificazione; insomma pdftex ha le funzionalità di tex, etex, pdfetex.

Per tradizione l'interprete tex, benché le sue funzionalità siano un sottoinsieme di quelle di pdftex, continua ad essere incluso in ogni distribuzione del sistema TEX. Se un altro programma, comunque si chiami, svolge almeno tutte le funzioni del programma tex, esso ha diritto di chiamarsi o di far parte della collezione della distribuzione del sistema TEX. Questo programma, progenitore di tutti gli altri, continua ad essere la pietra di riferimento per tutto il sistema; oggi non viene quasi più usato, se non, appunto come pietra di paragone. Non va dimenticato che il suo creatore, Donald E. Knuth, l'ha congelato e non intende più modificarlo; si limita a correggere i pochi errori che ancora contiene, premiando con un assegno firmato di suo pugno chiunque gli segnali un vero errore. Non va in bancarotta per due motivi: (*a*) di errori ce ne sono pochissimi, e (*b*) chiunque riceva un assegno firmato da Knuth stesso, non lo incassa ma lo espone in cornice. Il numero di versione di tex contiene le prime cifre del numero π, e ad ogni modifica acquisisce una nuova cifra tendendo asintoticamente a π, cosa che non avverrà mai visto che π è irrazionale (o meglio, trascendente). Oggi il numero di versione è 3,1415926; visto che le poche correzioni a tex vengono immediatamente inserite anche in pdftex, questo numero lo si ritrova nella prima riga di ogni file `.log`, qualunque sia il programma effettivamente usato direttamente o indirettamente attraverso lo *shell editor*.

Ciò non toglie che esistano due modalità di composizione (non importa se sono due distinte modalità di funzionamento dell'unico programma: pdftex): quella che in uscita produce un file DVI e quella che produce invece un file PDF.

I file sorgente elaborabili con latex o con pdflatex sono in generale identici, o possono essere resi tali, nel senso che usano lo stesso mark-up LATEX. Per cui quando si parla di un file elaborabile con latex, si intende che è stato scritto facendo uso del metodo di mark-up tipico di LATEX, e che il file è elaborabile indifferentemente anche da pdflatex. Sarebbe desiderabile che i file con questo mark-up avessero l'estensione `.ltx`, invece che l'estensione `.tex`, ma per vari motivi questa convenzione non viene quasi mai rispettata.

A causa dell'uso di pacchetti di estensione molto specializzati, un file apparentemente conforme alla grammatica del mark-up di LATEX può essere elaborato

con latex, ma può risultare incompatibile per essere elaborato con pdflatex; può succedere anche il contrario. Ciò succede raramente, ma non è impossibile che accada; sarebbe quindi desiderabile evitare tali pacchetti di estensione, ricorrendo a pacchetti meno specializzati ma adeguati per la particolare funzione per la quale essi vengono invocati. Questa soluzione può essere usata nella stragrande maggioranza dei casi.

Talvolta l'incompatibilità fra latex e pdflatex non dipende dal file sorgente, ma dai file inclusi; ciò può accadere facilmente con i file che contengono delle immagini; è per questo che sono molto utili quei programmi accessori che consentono di trasformare una immagine da un formato all'altro; a questo proposito si faccia riferimento al capitolo 10.

Comando di sistema	Interprete	File di formato sorgente	File di formato compilato
tex	tex	`tex.ini`	`tex.fmt`
etex	pdftex	`etex.ini`	`etex.fmt`
pdftex	pdftex	`pdfetex.ini`	`pdftex.fmt`
eplain	pdftex	`eplain.ini`	`eplain.fmt`
latex	pdftex	`latex.ini`	`latex.fmt`
pdflatex	pdftex	`pdflatex.ini`	`pdflatex.fmt`

Tabella 4.1: Legame fra alcuni comandi di sistema, gli interpreti e i formati

La tabella 4.1 riassume in parte quanto è stato esposto. L'unica differenza, vista la moltitudine di formati da generare in pratica con un solo interprete, è che invece di usare dei file sorgente con estensione `.tex` o `.ltx`, si usano dei file di inizializzazione con estensione `.ini` che provvedono a far leggere all'interprete i file specifici per il formato e per impostare i parametri necessari all'interprete per produrre i formati giusti in relazione al particolare modo di funzionamento del programma di sistema che si desidera attivare. In pratica i primi quattro formati attraverso i file di inizializzazione leggono tutti il file `plain.tex` insieme ad altri file con possibili estensioni del linguaggio, e impostano i parametri interni dell'interprete affinché l'esecuzione dei programmi i cui nomi *non cominciano* con pdf producano l'uscita in formato DVI, mentre quelli il cui nomi *cominciano* con pdf producano l'uscita in formato PDF. Gli ultimi due casi si riferiscono al programma latex la cui uscita è in formato DVI e al programma pdflatex la cui uscita è in formato PDF. Entrambi i formati sono creati leggendo sostanzialmente lo stesso file `latex.ltx` ma con inizializzazioni diverse a seconda del formato di uscita di default.

4.10 Altri programmi del sistema TEX

Vale la pena ricordare in questo capitolo la presenza di altri 'fratelli' di LATEX e pdfLATEX. Essi hanno campi di applicazione in parte complementari a quelli di

LaTeX e spesso vengono installati di default quando si installa il sistema TeX.

Tuttavia questo paragrafo e il successivo possono essere tranquillamente letti in un secondo tempo; servono per capire quanto sia vasto il mondo di TeX, ma costituiscono più una curiosità che non un insieme di informazioni utili a capire, sia pure a livello introduttivo, la potenza della tipografica con LaTeX.

4.10.1 Plain TeX

Il mark-up semplice, 'plain' in inglese, usato nei primi quattro esempi della tabella 4.1 è quello formato dall'insieme di comandi predisposto dall'autore Knuth e con il quale, tra l'altro, egli continua a comporre tutti i suoi libri.

Come linguaggio di mark-up è molto meno strutturato di LaTeX, tanto che talvolta il suo insieme di comandi viene qualificato come 'prescrittivo', invece che 'descrittivo'; LaTeX ha un insieme di comandi di mark-up decisamente più descrittivo, ma non è completamente descrittivo; così il mark-up Plain ha un insieme di comandi di gran lunga molto prescrittivo, e solo qualche comando di tipo descrittivo.

I programmi che usano il mark-up Plain, semplice o esteso[9], con uscita DVI o PDF, oggi vengono usati poco, ma in qualche raro caso si incontrano dei file sorgente scritti da altri che possono essere composti ricorrendo a tex o a pdftex. In questo caso gli *shell editor* sono in grado di eseguire questo compito con la semplice pressione di uno dei vari pulsanti presenti nei vari menù, quindi l'utente 'normale' non ha bisogno di conoscere il mark-up Plain. Sappia tuttavia che l'insieme di comandi di LaTeX costituisce una enorme sovrastruttura sui comandi di Plain TeX e che quasi tutti i comandi di Plain sono usabili sotto LaTeX e generalmente producono lo stesso effetto.

Per la documentazione ci si può rivolgere direttamente alla pietra miliare costituita dal libro e manuale di Knuth, il TeXbook, [31], che non dovrebbe mancare dalla scrivania di qualunque utente avanzato di LaTeX.

4.10.2 I programmi estesi

I programmi etex, elatex, pdfetex e pdfelatex deriverebbero dalla generazione del formato mediante il programma etex; questo oggi è totalmente assorbito dentro pdftex, quindi quei programmi non esistono più; tuttavia se ne parla solo per descrivere le estensioni di etex introdotte in pdftex.

Come si vede dalla tabella 4.1 oggi tutti questi programmi sono prodotti creando il formato sempre con l'interprete pdftex e quindi sostituiscono completamente tutti i programmi con la 'e' citati sopra; chi usa i programmi della tabella usa sempre programmi estesi, anche se non usa nessuna delle estensioni.

[9] Sono estesi i mark-up che contengono la lettera 'e' prima di 'plain' o prima di 'tex'; l'estensione consiste nell'uso della sillabazione per molte lingue (valido solo per il mark-up discendente da `plain.tex`) e nella disponibilità dei comandi estesi descritti successivamente in merito all'interprete etex.

Insomma i programmi di oggi sono *downward compatible* con i rispettivi progenitori. Tutti i pacchetti di estensione che fino ad oggi sono stati sviluppati per TEX, LATEX e i loro 'fratelli' sono quindi usabili anche con i programmi estesi; invece a tutt'oggi sono ancora pochi i pacchetti di estensione che richiedano espressamente l'uso dei programmi estesi e non siano compatibili con i progenitori, ma stanno aumentando di numero rapidamente.

C'è da sperare, però, che tutti i pacchetti di estensione esistenti vengano via via aggiornati, non tanto per dotarli di nuove funzionalità, quanto per sfruttare le estensioni dei 'modi' di esecuzione che possono svolgere nativamente certe funzioni, che fino ad oggi non sono sfruttate oppure che sono simulate con sequenze di istruzioni TEX o LATEX.

Vale la pena di sottolineare un fatto che a prima vista potrebbe apparire come superfluo, ma non lo è; è vero che fino ad oggi i programmi generalmente distribuiti su CTAN non sfruttano le nuove potenzialità o funzionalità dei programmi estesi, anche se questi sono i motori veri di ogni variante dei programmi di composizione del sistema TEX. Tuttavia questi programmi cominciano ad apparire; il primo che lo scrivente ha incontrato è proprio formato dalla classe `arstexnica` predisposta per comporre gli articoli della rivista ArsTEXnica dell'associazione GUIT. Questa classe di composizione richiede la modalità estesa dei programmi di composizione; di default, invece, l'aggiornamento di MacTEX del 2008, che lo scrivente aveva installato, predisponeva il sistema per usare, sì, i motori di composizione estesi, senza però attivare la modalità estesa. La situazione si è risolta ricostruendo tutti i file di formato in modalità estesa dando dalla finestra, o terminal, con i privilegi dell'amministratore il comando:

```
fmtutil -all
```

che ha, appunto, ricreato indistintamente tutti i file di formato in modalità estesa. Forse pochi lettori di questo testo si troveranno nella necessità di fare altrettanto, ma è bene sapere che, all'occorrenza, questa breve e semplice azione risolve tutti i problemi. Nelle versioni successive, dal 2009 in poi, il problema non si è più presentato.

Ma una cosa è quella di creare i file di formato che sono sinonimi dei 'programmi' da lanciare; un'altra è quella di sapere in quale modalità sta operando il motore di composizione; finora questo motore di composizione è stato chiamato *interprete*, perché il suo scopo è quello di interpretare le definizioni delle macroistruzioni per tradurle in linguaggio macchina, al fine di produrre un file di uscita contenente il testo tipocomposto.

Se con questo interprete si generano i file di formato avendo specificato `\pdfoutput=0`, il lavoro di composizione si concluderà con un file di uscita in formato DVI; al contrario se viene specificato `\pdfoutput=1`, il lavoro di composizione termina con un file in formato PDF. Il valore specificato per il parametro `\pdfoutput` viene integrato dentro il file di formato, per cui l'esito della composizione è predefinito, a meno che il compositore non voglia egli stesso specificare all'inizio del suo file sorgente un diverso valore del parametro `\pdfoutput`.

Ognuno può esaminare il proprio *shell editor* per vedere quale programma viene lanciato quando si clicca sull'icona che rappresenta LaTeX. Per chi scrive, che usa TeXShop su una macchina Mac, il bottone "Typeset" corrisponde a lanciare l'interprete pdftex in modalità estesa con il file di formato prodotto con `\pdfoutput=1`, quindi egli produce di default, in modalità estesa, file di uscita in formato PDF. Con lo shell editor LED su una macchina Windows, invece, il bottone "LaTeX" lavora con pdftex in modalità estesa ma produce in uscita un file DVI perché il formato `latex.fmt` è stato creato con `\pdfoutput=0`; quando si preme il bottone "pdfLaTeX" si lancia lo stesso interprete ma si lavora con un altro formato creato con `\pdfoutput=1`.

Generalmente oggi i file di formato sono tutti o quasi tutti composti in modalità estesa, o lo si può fare come si è descritto sopra. Tuttavia per trarne il miglior beneficio bisogna documentarsi a fondo nel file `.../doc/etex/base/etex_man.pdf`.

Ma in che cosa consiste questa modalità estesa? Permette di superare la limitazione di 256 contatori, scatole, lunghezze rigide, lunghezze elastiche, eccetera, che erano uno dei limiti del programma tex. Permette di comporre il testo sia con alfabeti 'normali' (da sinistra a destra) sia con alfabeti retrogradi (da destra a sinistra). Permette di eseguire alcuni semplici calcoli con i contatori e le lunghezze direttamente all'interno di espressioni speciali, ma facendo uso, sostanzialmente delle stesse notazioni operazionali che siamo soliti usare quando scriviamo l'algebra. Permette di eseguire alcuni altri test non previsti da tex, come per esempio verificare se un particolare segno esiste in un determinato font. Si può approfondire nel documento indicato sopra, `.../doc/etex/base/etex_man.pdf`, anche se si riferisce ad un programma che non esiste più, ma le cui funzionalità sono state trasfuse tutte in pdftex.

Secondo lo scrivente uno dei vantaggi della modalità estesa consiste anche nella possibilità di lanciare altri programmi durante l'esecuzione della composizione e questo, per esempio, consente di convertire al volo il formato delle immagini come si descriverà meglio nel capitolo 10 e, specialmente, per gestire gli indici analitici come esposto nel paragrafo 12.5.

4.10.3 Il mark-up ConTeXt

Il mark-up ConTeXt assomiglia molto vagamente sia a Plain, sia a LaTeX. Esso è un mark-up molto strutturato, ma tutto sommato semplice come Plain; la differenza sta nel fatto che virtualmente ogni comando accetta che vengano specificate delle opzioni per cui la personalizzazione della composizione avviene in modo più semplice, molto più semplice rispetto a quanto sarebbe necessario fare con Plain.

Esso permette di descrivere il testo da comporre in modo molto più strutturato anche rispetto a LaTeX, e i libri e gli altri documenti composti con questo programma sono generalmente molto ben riconoscibili per l'altissima qualità del risultato finale. Tutto ciò avviene grazie alle opzioni previste e attivate dai vari comandi usati per la descrizione del testo. Gli autori hanno spesso affermato

che essi preferiscono la libertà di definire qualunque dettaglio compositivo ma non volevano perdere le funzionalità offerte da LaTeX; direi che ci sono riusciti magnificamente e i risultati si vedono. Tuttavia il mark-up è così differente da quello di LaTeX che la maggior parte degli utenti continua a usare LaTeX.

ConTeXt inizialmente aveva come motore di composizione pdftex e quindi contiene le sue estensioni e i suoi limiti; il principale è quello di essere limitato a font che non contengano più di 256 caratteri, quindi non può lavorare con gli OpenType né con i font in codifica UNICODE. Questa versione di ConTeXt si chiama ConTeXt Mk II.

Ora (2013) è operativa una versione chiamata ConTeXt Mk IV che supera questi limiti; questa nuova versione si appoggia all'interprete LuaTeX, e a quello si rimanda per ulteriori informazioni.

4.10.4 I programmi Omega e Lambda, Aleph e Lamed

Il programma Omega, il cui sviluppo sembra si sia arrestato, cerca di sostituire il programma TeX in modo da togliergli tutte le limitazioni che ha, per esempio quella di trattare font con non più di 256 segni, e di arricchirlo di altre funzionalità. Tra le altre funzioni consente anche la composizione con alfabeti retrogradi; con l'arabo, che è sostanzialmente ancora una scrittura calligrafica molto legata, consente di eseguire legature molto complesse che coinvolgono anche quattro o cinque segni. In altre parole è decisamente più esteso di pdftex.

Il programma Lambda è, per così dire, la versione LaTeX di Omega; ne accetta tutte le estensioni e mette in grado il compositore di riferirsi ad una sintassi simile a quella di LaTeX.

Il programma Aleph è una versione estesa di Omega che permette di liberarsi di certi vincoli sui font usabili; per un certo periodo anche ConTeXt poteva usare Aleph come motore di composizione. L'autore di Aleph, Giuseppe Bilotta, ha lavorato anche al programma Lamed, versione di Aleph LaTeX-compatibile. Sembra però che anche questo programma sia stato assorbito in parte da luatex, e quindi il suo sviluppo ulteriore sia stato interrotto. Non deve però essere sottovalutata l'importanza di Aleph, perché rappresenta il passaggio da pdftex a luatex. Molte sue librerie sono state infatti incorporate in luatex.

4.10.5 Il programma XeTeX

Il programma XeTeX[10], ora disponibile per tutte le piattaforme di elaborazione e con qualunque sistema operativo, è una ulteriore estensione che riunisce molte delle funzionalità dei programmi precedentemente descritti, ma in più consente

[10] XeTeX è scritto in modo palindromo (simmetrico rispetto alla mezzeria) e lascerebbe pensare che si debba pronunciare anche in modo palindromo; il suo creatore, Jonathan Kew, ha generato il nome pensando a *eXtended ε-TEX* e quindi la prima 'X' non è una 'χ' maiuscola, ma è davvero una 'X'. Il nome dovrebbe quindi essere pronunciato 'xetekh'; il suo creatore lo pronuncia all'inglese 'zitek', ma a differenza di D. Knuth, non ne impone una pronuncia particolare.

di usare tutti i font presenti sul particolare calcolatore dove può essere impiegato, purché questi siano font di tipo OpenType, oppure TrueType oppure PostScript. Non occorrono estensioni particolari per l'uso di questi font, perché il programma sfrutta i comandi del sistema operativo per gestire i font disponibili e quindi una delle operazioni relativamente meno semplici da fare con LaTeX per gestire i font viene superata in un solo colpo. Il programma sta diffondendosi con rapidità specialmente in certi ambiti disciplinari; si tenga presente che gestisce tutti i possibili font di qualunque alfabeto, anche i sistemi di ideogrammi e gli alfabeti retrogradi come quello ebraico o quello arabo. Mescolare lingue diverse scritte con sistemi di caratteri completamente diversi diventa quindi una operazione semplicissima.

Ovviamente, per raggiungere questi scopi deve poter usare sia in entrata, sia in uscita la codifica UNICODE, anche se ha delle funzioni che gli consentono di usare i font vettoriali del sistema TeX, benché non siano codificati in UNICODE in senso stretto.

Esiste anche la versione LaTeX di XeTeX, chiamata XeLaTeX, e non è particolarmente difficile usarla pur di riferirsi al particolare manuale che l'accompagna, se non altro per prendere nota delle differenze rispetto al sistema LaTeX standard.

Il passaggio dalla compilazione con LaTeX a quella con XeLaTeX è praticamente indolore; se non si richiamano font diversi da quelli standard del sistema TeX, lo stesso file sorgente può venire composto sia con pdflatex sia con xelatex e si ottengono gli stessi risultati. I font usati di default sono quelli della collezione Latin Modern; infatti, invocando il pacchetto *fontspec* senza specificare nessun font particolare, viene usata la versione OpenType dei font che fanno parte della collezione Latin Modern. Si può usare anche il pacchetto *unicode-math* per usare le particolarità delle polizze di font che contengono anche i segni matematici codificati con la codifica UNICODE e si dispone di una scelta sterminata di segni matematici (oltre ad una miriade di altri segni) che possono mettere nell'imbarazzo della scelta. Anche altri programmi di impaginazione possono accedere ai font matematici UNICODE, ma solo il motore xetex riesce a gestirli in modo impeccabile, come fa anche pdflatex, rispetto a qualunque altro impaginatore o word processor.

4.10.6 Il programma LuaTeX

Il programma luatex è già nato, e nel 2012 ha raggiunto la versione 0.70.2 ed è abbastanza stabile; perciò luatex è già usabile oggi, ma in condizioni controllate, perché è ancora soggetto a molte modifiche. Questa volta si tratta veramente di un nuovo programma, che accoglie in sé diverse librerie C di pdftex (che a sua volta contiene sia tex sia etex), diverse di Aleph (che già contiene Omega), diverse di METAPOST, diverse di kpathsea, diverse del linguaggio di scripting lua, alcune delle quali sono state scritte apposta, diverse del visualizzatore xpdf, alcune dell'editor di font FontForge.

Il file sorgente di un documento da comporre segue il mark-up di ConTeXt mk IV e contiene sia il testo da comporre, sia le istruzioni per comporlo, ma

l'interprete luatex è in grado di svolgere anche altre funzioni che, per esempio, l'interprete pdftex non è capace di eseguire. Per esempio, gli interpreti classici del sistema TEX non sono in grado di eseguire agevolmente calcoli con operandi in notazione *floating point* tipica di ogni programma di calcolo; per questi interpreti un numero scritto nella forma `-0.12345E02` è privo di senso; pdftex è in grado di usare il numero decimale `-12.345` (equivalente a quello indicato in notazione floating point) solo come fattore di scala per le lunghezze. Per poter eseguire calcoli di ogni genere con numeri decimali e/o floating point bisogna ricorrere a macro specializzate con pdftex, mentre con luatex i calcoli si possono fare tranquillamente. Ciò è possibile per il fatto che luatex contiene un certo numero di librerie lua, compresa la libreria matematica, cosicché con i comandi di lua eseguibili all'interno di luatex si possono generare valori floating point già frutto di elaborazioni, e passare questi numeri ai moduli di composizione di pdftex per usarli nel modo che il sottostante linguaggio PDF possa gestirli nel modo richiesto per l'output.

Questo stesso principio vale per ogni altra caratteristica del testo da comporre visto che le librerie di lua sono in grado di eseguire ogni operazione eseguibile da un calcolatore. Perciò luatex è in grado di elaborare file da comporre con alfabeti strani, di eseguire ogni sorta di comandi condizionali in modo che lo stesso file sorgente possa produrre l'uscita in diversi formati, possa gestire i font OpentType testuali e matematici in modo relativamente semplice, eccetera. Una delle cose difficili da fare con pdftex, se non impossibili, ma facili da fare con luatex è quella di caricare i pattern di sillabazione al momento dell'esecuzione; un'altra è quella di generare al volo font virtuali e di crearne le relative informazioni metriche da usare durante la stessa esecuzione del programma. Grazie alle librerie di Aleph è possibile gestire le complicatissime legature, che possono coinvolgere anche quattro o cinque glifi, con alfabeti come l'arabo, che è molto calligrafico, ma anche come il font Zapfino, uno script "latino" fortemente calligrafico come appare nella figura 2.2.

Informazioni dettagliate si possono trovare nel sito del progetto LuaTEX: `http://www.luatex.org/`. Siccome è già stata presentata le versione 'beta', c'è da aspettarsi che LuaTEX diventi presto la soluzione di default per tutti o quasi i compiti di composizione tipografica. Un bel documento che traccia la storia dello sviluppo di ConTEXt e illustra le potenzialità di luatex si può scaricare da `http://www.pragma-ade.com/general/manuals/mk.pdf`.

Oggi, 2013, un file sorgente contenente un preambolo che sappia riconoscere quale sia il programma che si sta usando per comporlo e usi un preambolo conforme con quel programma, può venire compilato con i tre programmi principali descritti sopra se non si ricorre a funzioni speciali sui font e, soprattutto, senza ricorrere al linguaggio Lua. I tempi di esecuzione sono diversi; il più veloce è pdflatex e il più lento è lualatex, con xelatex che si colloca in posizione intermedia. Vale la pena di notare che pdflatex e lualatex permettono di usare la microgiustificazione completa, mentre xelatex ne

può sfruttare solo la *protrusione* nei margini. Perciò il risultato della composizione è leggermente diverso nei tre casi, anche perché lualatex e xelatex possono usare i font OpenType, mentre pdflatex di fatto usa solo i font Type 1. Quindi, quale dei tre programmi di composizione usare? La domanda non ha una risposta definitiva: dipende evidentemente da ciò che si vuole comporre.

4.11 Il sistema TEX

A conclusione di questo capitolo si spiega perché in questo testo si parla di "sistema TEX". Ormai dovrebbe essere chiaro, ma vale la pena di scendere in ulteriori dettagli.

Un sistema è un insieme coordinato di programmi che permettono di eseguire operazioni semplici o complesse in un ambito vasto. Di solito non sono composti di un unico programma o applicazione o eseguibile, comunque lo si voglia chiamare. Di solito è modulare in modo che per ogni operazione siano usate solo le risorse che sono utili, cioè solo i moduli funzionali per quella operazione. Questo è molto più efficiente di un unico immenso programma che faccia tutto, ed è molto più facile da rivedere per eliminare gli inevitabili piccoli errori o per aumentare le funzionalità complessive.

Il lettore curioso può esplorare il proprio disco rigido, dove ha installato una distribuzione del sistema TEX; troverà una cartella `bin/`⟨*piattaforma*⟩`/` che contiene i programmi direttamente eseguibili dalla macchina corrispondente alla particolare ⟨*piattaforma*⟩, oppure degli script eseguibili attraverso programmi di interpretazione, quali Perl, Java, Python, Bash, e simili; le macchine Windows interpretano a livello di sistema operativo le procedure *batch* scritte in file con l'estensione `.bat`.

Sulla macchina che si sta usando, un Mac con la distribuzione MacTEX 2014, la cartella `bin/universal-darwin/` contiene i nomi di circa 400 file, costituenti altrettante funzionalità del sistema nel suo complesso. Sulle macchine Windows e Linux il numero di eseguibili e di script può essere maggiore o minore, ma non è molto diverso.

Non si vogliono elencare qui le funzionalità di ciascuno di questi 400 programmi; ma si citeranno quei pochi che l'utente si può trovare ad usare in prima persona; in ogni caso quei programmi così importanti di cui non si può ignorare l'esistenza. Tutti quelli che si elencano qui sono già stati nominati o verranno nominati nel seguito.

Merita ricordare che i programmi elencati qui sono presenti in qualunque distribuzione *completa* del sistema TEX. Bisogna però ricordare che la distribuzione MiKTEX di solito viene (imprudentemente) installata in forma ridotta e l'utente si affida troppo spesso alla capacità di questa distribuzione di auto-installarsi le parti mancanti; purtroppo alcuni utenti non attivano questa funzionalità, ma tutti al momento del bisogno possono trovarsi in una situazione in cui non hanno accesso alla rete, per cui queste auto-installazioni automatiche non possono

avere luogo. Questa è una ragione in più per eseguire l'installazione completa di MiKTEX, perché contiene tutti questi programmi.

biber, BIBTEX, bibtex8, bibtexu sono i programmi che permettono di gestire i database bibliografici (esistenti e predisposti a mano attraverso le funzionalità di alcuni *shell editor*, oppure con altri programmi specifici, come BibDesk e Jabref, non distribuiti con il sistema TeX) per estrarne i dati da riportare in una o più bibliografie, a seconda del documento e dello stile di composizione richiesto; biber gestisce anche i dati codificati in UNICODE o UTF-8; BIBTEX, come anche i suoi programmi affini, gestisce solo i dati codificati in una delle tante codifiche dove ogni carattere non facente parte del sotto-insieme ASCII a 7 bit viene codificato con 8 bit; per gestire la moltitudine di alfabeti basati su quello latino, che impiegano molti caratteri con diacritici sopra o sotto i segni di base, non bastano i 128 caratteri con l'ottavo bit posto a 1, e quindi esistono tante codifiche diverse a seconda dei caratteri che vi sono elencati.

makeindex e xindy sono programmi che permettono di gestire i dati raccolti dal programma di composizione per creare indici analitici, glossari, elenchi di acronimi e simili; il primo gestisce solo caratteri codificati a 8 bit; il secondo gestisce anche i caratteri UNICODE e permette di eseguire ordinamenti alfabetici anche per alfabeti complessi come quello del greco politonico.

dvips, dvipdfm, dvipdfmx, xdvipdfmx, ps4pdf, pstopdf, epstopdf, mptopdf servono per eseguire trasformazioni di formato per i file di uscita dei programmai di composizione, o per trasformare le immagini dal formato 'encapsulated PostScript' in formato 'portable document format' conservandone la natura vettoriale; oppure per trasformare i file prodotti da METAPOST, in file nel formato 'portable document format'.

METAFONT insieme a TEX, questo è il programma creato da Donald E. Knuth per produrre i font che il sistema TeX ha usato fin dai suoi albori; esso agisce su file di descrizione della collezione di segni del font scritti in un linguaggio particolare e salvati in file con estensione `.mf`; METAFONT elabora questi file e produce i file metrici necessari al programma di composizione per comporre il testo, nonché i file bitmapped che contengono in un formato particolare le matrici di pixel che formano il disegno di ogni segno; i file bitmapped servono per presentare il risultato della composizione sullo schermo oppure per stamparlo su carta; agli albori di TEX erano praticamente indispensabili; successivamente sono stati sviluppati i font Type 1, TrueType e OpenType e la loro necessità è virtualmente svanita. Tuttavia questo programma viene invocato automaticamente ogni volta che il programma di composizione o di trasformazione del formato non trova i file metrici oppure non trova i file bitmapped o quelli vettoriali necessari per presentare il documento in una forma leggibile sullo schermo o sulla carta. Talvolta l'esecuzione di questo programma viene richiesta a

causa di un errore ortografico nei nomi dei font da usare oppure a causa di una installazione incompleta del sistema TEX; in questi casi METAFONT non riesce a portare a termine il suo compito; tuttavia, oltre a scrivere opportuni messaggi nel file `.log`, produce anche un file `missfont.log`, che l'utente trova nella stessa cartella del documento che sta componendo; leggendo questo file l'utente può capire di quali font mancanti si tratti e può provvedere in merito.

METAPOST è un programma di disegno programmato che permette di eseguire disegni descritti mediante una forma semplificata del linguaggio PostScript; l'utente usa sostanzialmente lo stesso linguaggio di METAFONT, con poche importati ma ovvie modifiche dovute allo scopo dei disegni che produce, ma i comandi per disegnare sono sostanzialmente gli stessi. Uno dei vantaggi di METAPOST è che i suoi disegni, salvati in file con una estensione numerica, possono essere inglobati nei documenti da comporre qualunque sia il programma che si usa per comporli. Generalmente è necessario cambiare l'estensione numerica in `.mps`, oppure impostare METAPOST perché inserisca il numero progressivo non nell'estensione del file, ma nel nome stesso del file, così da usare costantemente l'estensione `.mps`.

pdfcrop serve per scontornare le immagini in formato 'portable document format' in modo che, inserite nel documento, non solo appaiano nel punto desiderato, ma non siano racchiuse fra spazi bianchi eccessivi.

arara è un programma in grado di leggere le prime righe scritte in modo speciale in un file sorgente da comporre con uno dei programmi di composizione del sistema TEX; esso è capace di eseguire una serie di funzioni in modo tale da comporre il documento finito indipendentemente dal numero di compilazioni necessarie e dall'esecuzione di programmi esterni a quello di composizione.

latexmk è uno script esterno che provvede a gestire la composizione completa di un documento; in un certo senso è simile ad arara, ma è più difficile da personalizzare; siccome esiste da molti anni, certa documentazione fa ancora riferimento a latexmk invece che ad arara, perciò è opportuno conoscerne l'esistenza in modo da poter configurare arara adeguatamente per svolgere le operazioni specifiche che sarebbe stato necessario personalizzare mediante latexmk.

fmtutil, fmtutl-sys sono i programmi per generare o rigenerare i file di formato; raramente può accadere di doversene servire, ma succede.

updmap, updmap-sys sono i programmi necessari per costruire gli elenchi dei font disponibili, sia bitmapped sia vettoriali, ciascuno con le sue particolarità, affinché i programmi di composizione o di trasformazione la cui uscita è in formato PostScript o in formato PDF conoscano la loro esistenza e li possano gestire ognuno secondo le sue particolarità.

mktexlsr serve per aggiornare i database dei nomi dei file facenti parte del sistema TEX; se questi database non sono aggiornati, il sistema TEX non riesce a trovare ed usare certi file importanti, cosicché i suoi programmi di composizione non possono completare il loro compito.

kpseaccess, kpsepath, kpsereadlink, kpsestat, kpsetool, kpsewhere, kpsewhich, kpsexpand sono i programmi specifici che il sistema TEX usa per accedere ai file elencati nei database dei nomi dei file generati con mktexlsr; anche l'utente li può usare dando gli appositi comandi nella finestra comandi (o prompt dei comandi); è molto utile sapersene servire e servirsene.

tex, etex, latex, pdftex, pdflatex, xetex, xelatex, luatex, lualatex sono alcuni programmi di composizione del sistema TEX di cui si è già parlato in questo capitolo; non sono gli unici; esiste anche arlatex per comporre in arabo, ptex per comporre in giapponese e nelle altre lingue orientali; mltex e mllatex servono per comporre documenti multilingue (questi programmi sono conservati per retrocompatibilità), aleph e lamed per comporre con font codificati UNICODE (anche questi conservati per retrocompatibilità; le loro funzioni sono state assorbite da xetex, xelatex, luatex, lualatex); context, e diversi altri programmi di composizione.

texdoc è il programma il cui nome viene usato come comando, da scrivere in una finestra comandi o prompt dei comandi, per leggere la documentazione dei pacchetti o dei programmi; tale documentazione è già installata mediante la distribuzione prescelta, quindi evita di dover andare a cercare in rete ciò che è già disponibile nel proprio disco dove è installato il sistema TEX.

asy è il nome del file eseguibile del programma Asymptote; si tratta di uno strumento per disegnare con grafica vettoriale, un po' come METAPOST, ma secondo alcuni molto migliore; consente anche di eseguire disegni tridimensionali, cosa che METAPOST non può fare. Viene spesso suggerito come una alternativa rispetto ad eseguire disegni con altri metodi, come per esempio *pgfplots* oppure *TikZ*, che sono pacchetti di disegno programmato basati sul linguaggio LATEX.

Quasi tutti i programmi che fanno parte delle distribuzioni TEX sono documentati in inglese, alcuni, pochi, in polacco o in tedesco; la documentazione è leggibile con il programma texdoc. Quelli elencati sopra sono tutti documentati in inglese.

L'utente normale generalmente si serve di uno *shell editor* per elaborare i suoi file `.tex`; questi editor sono forniti di voci di menù a discesa o di icone da selezionare per eseguire molti dei programmi elencati sopra. Il processo è così trasparente per l'utente, che pensa che l'editor sia quello che fa tutto. Si raccomanda sempre di non confondere l'editor con i programmi che l'editor è capace di usare selezionando una voce da un menù o cliccando su una icona; queste funzionalità sono presenti nell'editor per la comodità dell'utente; alcune rare volte

queste funzionalità preconfezionate non soddisfano le necessità dell'utente, che deve quindi essere in grado di aprire un terminale (finestra comandi o prompt dei comandi) e scrivere gli opportuni ordini in questa finestra. È quindi opportuno che l'utente conosca l'esistenza di questi programmi e, in caso di bisogno, sappia documentarsi e li possa usare per soddisfare le proprie necessità.

Capitolo 5

LaTeX: prime nozioni

5.1 Introduzione

Verranno ora esposte le prime nozioni per usare LaTeX e per comporre semplici documenti di solo testo. Sarà necessario mostrare esempi di codice sorgente, così come sarà necessario descrivere la 'sintassi' di alcuni comandi.

Il codice e i vari comandi saranno composti con il font della famiglia a spaziatura fissa, per esempio si scriverà `\documentclass`. La grammatica prevede che ogni comando possa avere degli argomenti il cui significato logico viene indicato da un nome o da una semplice locuzione scritta in corsivo e racchiusa fra parentesi 'acute', per esempio ⟨*classe*⟩, così che la sintassi del comando `\documentclass` possa essere espressa così:

`\documentclass[`⟨*opzioni*⟩`]{`⟨*classe*⟩`}`

Con questo tipo di grammatica gli argomenti che obbligatoriamente devono essere forniti ad un comando che richieda argomenti sono sempre racchiusi fra parentesi graffe, mentre gli argomenti facoltativi, opzionali, se sono presenti, sono sempre racchiusi fra parentesi quadre. Le parentesi acute della descrizione della grammatica servono solo per questa descrizione e non vanno effettivamente introdotte quando il comando viene usato.

5.2 L'inizio del file sorgente

Il comando `\documentclass` è il primo che deve essere dato all'inizio di un file (o nel primo file di un gruppo) da trattare con LaTeX.

La ⟨*classe*⟩ indica il tipo di documento che si intende comporre; le classi standard del sistema sono *book*, per comporre libri, *report* per comporre rapporti 'tecnici', e *article* per comporre articoli; esistono ovviamente altre classi non standard, ma queste tre sono le più note sotto ogni aspetto.

Per questo libro è stato usato il comando

```
\documentclass[b5paper]{book}
```

e con questa dichiarazione iniziale si è detto che si vuole comporre un libro e che lo si vuole stampare su carta ISO-UNI B5 (176 mm × 250 mm).

L'inizio del file sorgente prosegue facoltativamente con una serie di specificazioni per rendere più agevole la scrittura del file sorgente, per scegliere la lingua o le lingue da usare nel documento, per scegliere la codifica di default dei font da usare per comporre il documento, per estendere i comandi di default con istruzioni non standard al fine di ottenere risultati particolari.

Questo insieme di dichiarazioni preliminari si chiama *preambolo*; non è necessario che il preambolo contenga qualche cosa, ma in pratica il preambolo in Italia e per i documenti in italiano contiene almeno la specificazione dell'uso della lingua italiana e probabilmente la specifica dei font a 256 caratteri che contengono le lettere accentate; questo vale praticamente in ogni paese, tranne quando la lingua da usare è solamente l'inglese che per LaTeX è la lingua di default e la sua scrittura non contiene lettere accentate. Per questo libro la prima parte del preambolo comincia (in forma semplificata) così:

```
\documentclass[b5paper]{book}
\usepackage[utf8]{inputenc}%%%%%%%%%%
\usepackage[T1]{fontenc}
\usepackage[italian]{babel}
\usepackage{guit}

\author{\GuIT}
\title{Introduzione all'arte\\
           della composizione tipografica con~\LaTeX}
\date{Versione 0.0}
```

Come vi si può leggere in chiaro, in questo preambolo si ordina a LaTeX di usare un certo numero di pacchetti di estensione mediante il comando `\usepackage`: in ordine gli si ordina di usare il pacchetto per gestire l'immissione del testo, specificandogli come opzione che i caratteri immessi corrispondono alla codifica *utf8*; poi gli si ordina di usare il pacchetto per gestire la codifica (encoding) per i font di composizione del documento mediante la sigla *T1*, che corrisponde alla codifica standard di LaTeX per poter usare i caratteri latini accentati (vedi il capitolo 26); poi gli si comanda di usare il pacchetto *babel* per gestire lingue diverse dall'inglese e come opzione gli si specifica l'italiano; poi infine gli si ordina di usare il pacchetto *guit* per poter avere a disposizione il comando `\GuIT` per comporre la sigla del Gruppo degli Utilizzatori Italiani di TeX; volendo si potrebbe usare anche il logo del gruppo.

I primi tre pacchetti sono pressoché *obbligatori* per ogni documento composto in italiano e anche, specificando un diverso nome della lingua, per comporre documenti in qualunque lingua diversa dall'inglese; si possono specificare anche diverse lingue mediante una lista di nomi (in inglese) separati da virgole e senza spazi inclusi, l'ultima delle quali è quella usata di default nel documento.

Alcuni suggeriscono di usare la codifica *utf8* per tutte le macchine e ogni sistema operativo; alcuni *latin1*; altri per le macchine Windows suggeriscono la codifica *ansinew*; altri ancora suggeriscono la codifica *latin9*; per le macchine Macintosh 'vecchiotte' (con sistema operativo precedente al Mac OS X) alcuni preferiscono la codifica *applemac*. Questa questione della codifica verrà sviluppata più avanti nel capitolo 26; per ora si preferisca la codifica *utf8* che va bene per tutte le macchine ed è adatta per scrivere in qualsiasi lingua.

Per il file di uscita l'opzione *T1* è senz'altro consigliabile per scrivere in italiano, direi quasi obbligatoria. L'alternativa è di non specificare nulla; la differenza passerà inosservata al principiante, ma dopo un po' di esperienza anche il neofita si accorge che senza specificare nulla, usando quindi la codifica predefinita *OT1*, le lettere accentate del file di ingresso o non appaiono per nulla, oppure appaiono con gli accenti non perfettamente centrati sulla lettera di base, e che le parole contenenti lettere accentate tendono a produrre una giustificazione peggiore delle righe a causa del fatto che la divisione in sillabe è meno efficiente con la codifica *OT1*.

Dopo la riga vuota (o bianca) si specifica che l'autore di questo libro è il GUIT stesso, si specifica il titolo del libro e, invece della data, si specifica il numero della versione. 0.0 è il valore iniziale, ma è probabile che il documento che state leggendo abbia un numero successivo, che non sia, cioè, la versione 'zeresima'.

Il preambolo di questo libro contiene altre cose che verranno descritte più avanti. Finito il preambolo comincia la composizione vera e propria del documento.

5.3 Il documento

Come è facile prevedere, il documento comincia con

```
\begin{document}
```

e, altrettanto prevedibilmente, finisce con

```
\end{document}
```

Tutto ciò che è racchiuso fra queste due righe viene composto o viene usato per controllare la composizione. È opportuno notare che tutto ciò che nel file compare dopo `\end{document}` non viene stampato ma può essere usato per scrivere dei commenti relativi al file, allo stato di avanzamento della composizione del documento, o ad altro che possa interessare il compositore per sua futura memoria.

In questo libro il documento comincia (nella versione 0.0 cominciava) così:

```
\begin{document}
\frontmatter
\maketitle
\tableofcontents
```

```
\chapter*{Presentazione}
```

dove il primo comando dichiara che si stanno componendo le pagine preliminari; il secondo comando serve per comporre la pagina del titolo usando le scarne informazioni circa l'autore, il titolo e la versione fornite nel preambolo; il terzo comando ordina di comporre l'indice generale; la riga bianca non conta niente, ma ha un effetto visivo di separazione nel file sorgente; in generale essa serve per marcare la fine di un capoverso, ma in questo punto non ci sono capoversi di nessun genere da comporre.

Il comando `\chapter*` serve per comporre un capitolo; esso non viene numerato finché si compone la parte iniziale del documento ma potrebbe essere incluso il suo titolo nell'indice generale, ma la presenza dell'asterisco lo impedisce; titoli dei capitoli generati con il comando asteriscato non compaiono nemmeno nelle testatine né nell'indice a meno che non li si voglia espressamente indicare mediante comandi appositi che per ora esulano da questa descrizione. L'argomento di questo comando indica il titolo da dare a questo capitolo.

Come si vede il mark-up contiene un'unica piccola difficoltà per coloro che non hanno nessun rudimento di inglese, cioè che tutti i comandi sono in inglese; il buono è che, non contenendo abbreviazioni, questi comandi sono immediatamente comprensibili senza bisogno di ricorrere a elenchi di comandi criptici.

5.4 La fine del documento

Quando il testo del documento è stato completamente introdotto nel file sorgente, non resta che dichiararne la fine chiudendo l'ambiente *document* mediante il comando

```
\end{document}
```

Dopo questa riga e questa istruzione, si può scrivere quello che si vuole, ma non verrà mai composto; questa è la dichiarazione che il documento è terminato, non che il file è terminato.

5.5 Un semplice esercizio

Esercizio 5.1 Copiate dal testo precedente i comandi illustrati a partire dal comando `\documentclass`; dopo `\chapter*{...}` introducete un qualunque testo con capoversi sufficientemente lunghi, magari un capoverso in italiano, uno in inglese, uno in francese (non dovrebbe essere troppo difficile trovare dei libri o dei giornali che contengano delle frasi, anche in lingua straniera, da copiare, se la propria fantasia e la propria conoscenza delle lingue straniere non fosse sufficiente). Alla fine del documento ricordatevi di dichiararne la fine; sembra lapalissiano, ma talvolta ce ne dimentichiamo.

Salvate il file scritto rigorosamente in formato testuale (meglio se facendo uso di uno *shell editor* adatto all'uso di LaTeX) per esempio con il nome `esempio1.tex` (il nome non è importante, ma l'estensione *deve* essere `.tex` (o, se proprio volete, `.ltx`).

Lanciate LaTeX; se state lavorando con uno shell editor[1], cliccate sull'opportuna icona; se non state lavorando con un shell editor aprite la finestra comandi o terminal o xterm o console, comunque si chiami sul vostro calcolatore quella finestra nella quale potete inserire i cosiddetti comandi in linea, scrivendo esplicitamente i nomi dei programmi da lanciare e del file su cui devono operare; in questo esempio scriverete

```
latex esempio1
```

senza specificare l'estensione `.tex` se aveste usato questa estensione, oppure esplicitando l'estensione diversa da `.tex` che avete usato per il vostro file.

Se nella finestra comandi o nell'opportuna finestra dello shell editor compaiono messaggi d'errore, rivedete il testo perché gli unici errori che potete avere fatto consisterebbero in errori di battitura; rivedete con particolare attenzione di aver usato le parentesi graffe dove ci vogliono le graffe, le parentesi quadre dove ci vogliono le quadre, e che tutte le parentesi si accoppino correttamente, cioè che ad ogni parentesi aperta segua una parentesi chiusa dello stesso tipo, come in matematica; quindi: aperta–chiusa, aperta–chiusa,... oppure: aperta, aperta – chiusa, chiusa. Questo è uno degli errori più comuni; inoltre i comandi devono essere scritti correttamente: rivedetene l'ortografia ricordandovi che il correttore ortografico del vostro editor di solito non rileva e non corregge gli errori di battitura dei comandi.

Se le ultime righe di quel che compare nella finestra dicono che il file è stato correttamente salvato nel file `esempio1.dvi`[2] lanciate il previewer adatto a quel formato; potrà essere YAP se avete il sistema MiKTeX; sarà xdvi se state operando su Linux, eccetera.

Ammirate la vostra composizione sullo schermo e osservate se il testo in inglese e/o in francese è stato composto bene, in particolare se le cesure in fin di riga sono corrette.

Questo semplice esercizio vi permette di rilevare quanto sia semplice usare LaTeX, specialmente se state usando uno shell editor; con quest'ultimo potete premere un solo pulsante che vi attiva l'esecuzione di LaTeX e, se non ci sono errori, vi attiva anche il previewer; ad ogni successiva compilazione generalmente la finestra del previewer viene automaticamente aggiornata, per cui avete una

[1] D'ora in poi la locuzione inglese *shell editor* nn sarà quasi mai scritta in corsivo, ma la si considererà acquisita come locuzione nomale per questo testo.

[2] Se state usando TeXShop su un Mac, il file viene salvato in formato PDF e il previewer viene aperto automaticamente.

compilazione quasi sincrona; vi consiglio di compilare sovente e di verificare sovente sul previewer che non ci siano errori grossolani di composizione, per esempio potreste aver dimenticato l'effetto di una dichiarazione e quindi potreste scoprire che da un certo punto in avanti state componendo con un font, perché vi siete scordati di ripristinare il font precedente. Questi sono errori grossolani che conviene correggere prima di avere composto le settecento pagine del libro che volete scrivere...

Gli errori più fini vanno ricercati nelle successive revisioni delle bozze, che, per fortuna, non richiedono di stampare su carta una serie di bozze dietro l'altra, ma possono essere riviste direttamente sullo schermo.

Va notato che molti sistemi TeX permettono di eseguire sia la *forward search* sia la *reverse search*. Queste operazioni prevedono una interazione fra l'editor e il previewer in modo che azionando opportuni pulsanti dell'editor mentre il cursore si trova in una certa posizione nel testo del file sorgente, nella finestra del file compilato il cursore viene collocato se non nella stessa posizione, qualche parola più avanti o più indietro; questa è la *forward search*. Analogamente con il cursore nella finestra del previewer in una certa posizione, cliccando su un apposito pulsante o premendo una opportuna combinazione di tasti, il cursore si sposta nella finestra dell'editor praticamente nella stessa posizione, parola più, parola meno; questa è la *reverse search*. Grazie a questa interazione fra editor e previewer la correzione delle bozze procede spedita in quanto la finestra dell'editor e quella del previewer sono praticamente sincronizzate.

È probabile che le parti di testo scritte in lingue diverse dall'italiano risultino avere alcune parole divise in sillabe in fin di riga (la cesura) in modo errato; questo dipende dal fatto che nel preambolo dell'esempio si è invocato il pacchetto per il trattamento delle lingue e gli si è specificato di voler comporre in italiano. È ovvio che la sillabazione delle altre lingue è diversa da quella dell'italiano, ed è altrettanto ovvio che scrivendo in lingue diverse dall'italiano è possibile che alcune cesure risultino errate.

Avete probabilmente usato, specialmente per il francese, molte lettere accentate; avrete quindi sfruttato a fondo le possibilità del vostro editor per inserire i caratteri accentati diversi da quelli che vi compaiono già sulla vostra tastiera, come per esempio 'ô' di 'diplôme'; molto probabilmente il file composto visualizzato nel previewer vi mostrerà il segno giusto.

5.6 I caratteri speciali

Per scrivere il mark-up secondo LaTeX bisogna ricordare che alcuni caratteri hanno un significato speciale; essi sono i seguenti[3]:

```
\   {   }   %   $   ^   _   &   #   ~
```

[3] In verità ci sarebbero anche i caratteri `‘, ’, "` che in certe circostanze svolgono funzioni speciali; il lettore non se ne preoccupi qui, ma sappia che in italiano il doppio apice svolge moltissime funzioni, tranne quello di doppie virgolette diritte.

Il carattere \, chiamato *backslash* in inglese e *barra inversa* in italiano, serve come iniziatore di un comando o istruzione, come lo si preferisce chiamare; una istruzione è sempre costituita dal segno \ seguito senza spazi interposti da una *stringa di lettere minuscole e/o maiuscole* e questa stringa termina quando è seguita da un carattere non alfabetico; oppure una istruzione è formata dal carattere \ seguito da *un solo carattere non alfabetico*; oppure essa è costituita da un carattere *attivo*. Nella lista di caratteri speciali illustrata sopra il carattere ~ è un carattere attivo, ed è l'unico carattere attivo definito da LaTeX. Se ne possono avere altri, ma questo si può ottenere solo con pacchetti di estensione; per esempio con l'opzione per la lingua italiana di `babel`, il carattere " è anch'esso attivo, come lo è per quasi tutte le lingue con la notevole eccezione dell'inglese.

I caratteri { e } sono molto particolari perché servono per delimitare gli argomenti dei comandi, ma svolgono anche altre funzioni di cui si parlerà in seguito.

Il carattere % svolge il compito di inizio di un commento: come in tutti i linguaggi di programmazione esiste una maniera per inserire dei commenti in linea con il codice, così anche con il sistema TeX l'inserzione del carattere % in una riga ordina al motore TeX di trascurare tutti i caratteri che seguono % sulla stessa riga, fino al primo carattere diverso da uno spazio (noto anche con il nome di *blank*) nella riga successiva.

Il carattere $ serve come 'interruttore' per ordinare a TeX di passare dal modo testo al modo matematico o viceversa. Quando l'interprete TeX è in modo testo interpreta i caratteri del file d'entrata in una certa maniera, mentre quando è in modo matematico, li interpreta come segni matematici. Per esempio, lo stesso carattere - in modo testo indica un trattino '-', mentre in modo matematico rappresenta il segno negativo '−', e si vede chiaramente che si tratta di due segni diversi; un analogo diverso modo di interpretare i caratteri del file di entrata viene usato da TeX per gestirli appropriatamente quando esso è in modo matematico.

Il segno ^ viene usato solo in matematica e indica che quanto segue eventualmente racchiuso fra parentesi graffe, rappresenta un esponente o un apice. Per usarlo come accento circonflesso nel testo basta farlo precedere da un backslash; per indicare un accento circonflesso in modo matematico bisogna usare il comando `\hat`.

Il segno _ serve in matematica per indicare un deponente o un pedice. Per usarlo come 'underscore' sia nel testo sia in matematica basta farlo precedere dal backslash.

Il segno & serve per separare fra di loro i brevi testi o le brevi espressioni matematiche che entrano nelle celle di una tabella o di una matrice. Preceduto dal backslash serve per comporre l''e commerciale', *ampersand* in inglese.

Infine il segno # serve per marcare il numero progressivo dell'elenco dei parametri nella definizione di una nuova istruzione; preceduto dal backslash serve per comporre il *number sign* oppure *hash sign*, in italiano chiamato gergalmente 'cancelletto'.

Per scrivere questi caratteri in un brano di testo senza che svolgano il loro compito normale, bisogna farli precedere dal backslash. Ecco quindi che per esprimere un ammontare di denaro in dollari bisogna usare `\$` nel file sorgente per cui `\$234,99` diventa $234,99. Analogamente per indicare un incremento del 10% bisogna scrivere `10\%` nel file sorgente. L'unico carattere che non è stampabile ricorrendo al segno `\` è proprio questo stesso segno `\`. Per stamparlo bisogna ricorrere al comando `\textbackslash`. Siccome nel testo questo segno non si usa quasi mai, il fatto di dover ricorrere ad un comando così lungo non costituisce un vero problema. Generalmente sono disponibili anche i comandi `\textasciicircum` e `\textunderscore` che servono per inserire i rispettivi segni durante la composizione del testo. Esistono anche i comandi `\textdollar` e `\textasciitilde`, quest'ultimo per inserire una tilde invece di uno spazio non separabile, come avviene usando il carattere attivo `~`.

5.7 Testo strutturato

Questo paragrafo è preliminare a quello che segue, ma merita un suo spazio autonomo, perché strutturare un testo è una operazione che va fatta sempre, anche con i documenti più semplici.

Persino una lettera commerciale è strutturata in campi disposti variamente su uno o più fogli; lo spazio per il nome e l'indirizzo del destinatario, quello per il nome e l'indirizzo del mittente, i riferimenti d'archivio, l'oggetto della lettera, l'apertura, il testo, la chiusura, l'eventuale elenco degli allegati, l'elenco dei destinatari in copia, eccetera. Ognuna di queste strutture può venire posta in riquadri o comunque in posizioni adeguatamente scelte dallo stile di scrittura preferito. Spesso le indicazioni del mittente sono prestampate sul foglio mediante l'uso di *carta intestata*; l'intestazione può contenere anche il logo dell'azienda e altre informazioni non elencate sopra.

Certo, una lettera ha la sua struttura e non è divisa in capitoli, paragrafi, sottoparagrafi, e simili. Altri documenti, invece, sono suddivisi con questi sezionamenti; altri ancora, come un dizionario, una Bibbia, un Codice Penale, sono fortemente strutturati, ma diversamente da come descritto per un comune libro.

Usando il sezionamento prodotto da capitoli, paragrafi, sottoparagrafi, bisogna stare attenti ad alcune regole generali che sarebbe bene osservare sempre, anche se talvolta sembra necessario fare delle eccezioni.

Ogni sezione dovrebbe contenere almeno due sezioni di rango inferiore perché queste ultime abbiano un senso; quindi non si divide in parti un documento che contenga un solo capitolo, tenendo conto che la divisione in parti è sempre facoltativa. Ma non si divide in paragrafi un capitolo che contenga un solo paragrafo, e via di questo passo. Perciò per un capitolo la divisione in paragrafi richiede sempre almeno due paragrafi; per ogni paragrafo sono sempre richiesti almeno due sottoparagrafi; eccetera. Nessuno vieta, naturalmente, di cominciare un capitolo con alcuni capoversi non preceduti da un sezionamento, ma aventi

carattere introduttivo, per poi cominciare il sezionamento con i paragrafi. Lo stesso vale per ogni livello di sezionamento.

Ma come ogni sezione deve contenere almeno due sezioni di rango inferiore, deve anche essere chiaro quando la sezione termina e ne comincia un'altra; non ha quindi senso cominciare un paragrafo e dopo alcuni capoversi aprire un sottoparagrafo; chiuso questo, riprendere li discorso del paragrafo, con alcuni capoversi, e poi aprire un altro sottoparagrafo. Procedere in questo modo fa perdere al lettore il senso della strutturazione. Perciò una volta cominciata la strutturazione con sezioni di rango inferiore, la si termina con l'apertura di una sezione di rango relativo superiore completa con il suo numero (se è richiesta la numerazione) e il suo titolo.

Inutile dire che non si possono saltare livelli di sezionamento; aperto il capitolo si procede con i paragrafi[4]; dentro i paragrafi si procede con i sottoparagrafi; dentro i sottoparagrafi, se davvero fosse necessario scendere a livelli così bassi, si procede con i sotto sottoparagrafi.

Quindi prima di sezionare, conviene programmare bene come segmentare il testo per non trovarsi poi con situazioni difficili da gestire al fine di mantenere la coerenza della strutturazione.

5.8 Organizzazione dei file sorgente

Questo paragrafo andrebbe scritto in fondo a questo testo, ma tutto sommato è logicamente connesso con il materiale esposto in questo capitolo. All'occorrenza potete saltarne la lettura, salvo poi tornare a questo paragrafo quando dovrete gestire la composizione di un lungo testo molto strutturato.

I semplici esercizi che si possono fare ora e con il contenuto dei primi capitoli che seguono, comportano la scrittura di brevi file sorgente che raramente, una volta compilati, superano il paio di pagine di testo composto.

Il file sorgente non richiede, quindi, una particolare organizzazione, perché la sua suddivisione naturale in 'preambolo' e 'corpo del documento' è più che sufficiente.

Tuttavia si immagini di dover scrivere un testo diviso in numerosi capitoli, appendici, indici e glossari, prefazioni e postfazioni, eccetera. Il file sorgente monolitico rischierebbe di diventare enorme, senza contare l'organizzazione dei file da includere, come i file che contengono le immagini, o i file che conviene predisporre a parte perché richiedono una paziente messa a punto del contenuto.

In questo caso l'organizzazione monolitica non è consigliabile, e infatti LaTeX offre diversi strumenti per spezzare il file sorgente in file di minori dimensioni. LaTeX, infatti, offre due insiemi di comandi per questo scopo: `\input` e `\include` assieme a `\includeonly`, oltre al comando `\includegraphics` per immettere nel flusso di informazioni da elaborare i file contenenti le immagini. Per quest'ultimo

[4] In questo paragrafo si è usata la parola *paragrafo* piuttosto spesso. Il lettore, naturalmente non avrà confuso il paragrafo con il *paragraph*...

comando si veda più avanti il capitolo 10. Per gli altri comandi la sintassi è la seguente:

```
\input{⟨file⟩}

\includeonly{⟨lista di file⟩}
\include{⟨file⟩}
```

L'informazione ⟨*file*⟩ è solamente il nome di un singolo file, mentre ⟨*lista di file*⟩ è un elenco di nomi di file separati da virgole, senza spazi prima o dopo le virgole. Per tutti i nomi dei file l'estensione `.tex` è sottintesa e quindi questa deve essere specificata se si vuole inserire un qualunque altro file con una estensione diversa, compresa l'estensione `.ltx` che in teoria dovrebbe essere l'estensione di default per tutti i file scritti con il mark-up del linguaggio LaTeX.

Merita invece osservare che quasi tutti i sistemi operativi ammettono che i nomi dei file possano contenere anche spazi e altri punti oltre a quello che separa l'estensione. Un nome di file del tipo `parte II.tex` oppure `parte.2.tex` è perfettamente lecito con moltissimi sistemi operativi. Ma è altrettanto ovvio che gli spazi e i punti, come anche gli altri caratteri che in TeX hanno un significato particolare, ma sono leciti nei nomi dei file, possono produrre guai seri. Si sconsiglia, per esempio, di usare la linea ribassata (*underscore* in inglese), "_", perché per TeX quel segno indica i pedici in matematica ed è illegale in modo testo. Oggi spesso questo segno viene gestito correttamente se è presente in un nome di file, tuttavia se ne sconsiglia l'uso, e se si desidera avere un "allontanamento" visivo delle parti del nome del file si preferisca il trattino normale `-`.

Però si evitino assolutamente i punti; se ci si riferisce al file `parte.2.tex` il ⟨*file*⟩ potrebbe essere `parte.2` ma TeX potrebbe scambiare `.2` per l'estensione e lamentarsi che non trova il file.

Si potrebbe racchiudere il nome del file fra doppi apici: `"parte II"` oppure `"parte.2"` ma c'è il problema che il doppio apice in quasi tutte le lingue invocate con *babel*, italiano compreso, è definito come carattere attivo, cioè è definito in modo che si comporti come una istruzione, non come un segno grafico; per questo motivo nel momento in cui si vuole leggere un file esterno mediante `\input` o `\include`, l'istruzione corrispondente al doppio apice viene eseguita e TeX si ritrova confuso e non sa che cosa leggere cosicché emette dei messaggi d'errore che talvolta sono difficili da associare al file che TeX non riesce a leggere.

Ci si ricordi infine che in molti sistemi operativi le lettere maiuscole e minuscole sono "diverse" al fine di identificare e distinguere i file. Per i sistemi UNIX e Linux i file `PartePrima.tex` e `parteprima.tex` sono due file diversi; nei sistemi Windows quei due nomi identificano lo stesso file.

Per evitare gli inconvenienti menzionati sopra si consiglia di usare nomi di file formati da una sola parola alfanumerica, senza punti e spazi inclusi, seguita dalla estensione opportuna, possibilmente scritti completamente con lettere minuscole. In questo modo non bisogna ricorrere a nessun espediente per gestire questi

nomi durante l'esecuzione di una compilazione, ma, ancora più importante, non si generano inconvenienti quando si scambiano i propri file con collaboratori che lavorano con altre macchine e/o con altri sistemi operativi. La *portabilità* è un bene prezioso che gli utenti del sistema TEX apprezzano moltissimo.

Il comando `\input` ordina a LATEX di leggere il ⟨*file*⟩ specificato; il flusso di dati che viene letto, viene inserito nel flusso complessivo del documento come se fosse inserito esattamente al posto di `\input`; che questo ⟨*file*⟩ contenga una sola parola, o un intero paragrafo o capitolo non ha nessuna importanza; il suo contenuto viene letto incondizionatamente ed elaborato immediatamente.

Questo comportamento del comando `\input` si mantiene anche se è contenuto dentro altri file a loro volta inseriti nel flusso di elaborazione dati mediante altri comandi `\input` o anche comandi `\include`; non esiste un limite teorico al numero di file che possono essere inclusi richiamandosi in catena, salvo la limitazione ovvia delle dimensioni finite della macchina di elaborazione e del numero di canali di ingresso che possono essere aperti contemporaneamente; questo numero massimo di canali è di sedici unità, ma alcuni di questi canali sono già impegnati per altre funzioni di servizio, per cui si può contare su una decina di canali effettivamente disponibili; questo numero dovrebbe essere sovrabbondante rispetto a qualunque esigenza insolita di immissioni a catena mediante comandi `\input`.

Il gioco di `\includeonly` e `\include` è invece più complesso; innanzi tutto, in assenza di `\includeonly` e della sua ⟨*lista di file*⟩, `\include` agisce quasi come `\input`; l'unica differenza è che prima di elaborare il contenuto del file incluso, LATEX ordina di eseguire un salto di pagina, cioè di iniziare una nuova pagina. Questo permette di capire subito che `\include` non è usabile al posto di `\input`, proprio a causa di questo salto di pagina. Il comando `\include` è più adatto per includere un intero capitolo (che comincia sempre su una pagina nuova) piuttosto che per immettere semplicemente del testo da elaborare nel punto in cui compare il comando.

Ma tutto diventa più interessante e utile se si fanno agire `\includeonly` ed `\include` assieme; infatti in questo modo `\include` include solo i ⟨*file*⟩ che compaiono nella ⟨*lista di file*⟩; però, se l'argomento di `\include` è un ⟨*file*⟩ che non compare nella ⟨*lista di file*⟩, `\include` ordina a LATEX di leggere solo il file `.aux`, cioè il file ⟨*file*⟩`.aux`, che contiene le informazioni relative ai riferimenti incrociati e al contenuto dei vari contatori che interessano LATEX, cosicché anche se il file ⟨*file*⟩`.tex` non viene elaborato, le sue informazioni utili sono tenute in conto e tutto procede agli effetti esterni come se il file fosse stato elaborato. Va da sé che se questo file ⟨*file*⟩`.tex` non è mai stato elaborato prima, non esiste nemmeno il file ⟨*file*⟩`.aux`, ma `\include` non protesta perché si limita a scrivere un avvertimento nel file `.log` privo di qualunque conseguenza sul buon fine della compilazione.

Un file incluso mediante `\include` può contenere a sua volta comandi `\input`, i file dei quali a loro volta... Va da sé però che, se un file non viene incluso perché il suo nome non compare nella ⟨*lista di file*⟩, non vengono elaborati nemmeno gli eventuali file subalterni richiamati dai comandi `\input`.

Al contrario di `\input`, i comandi `\include` non possono essere innestati l'uno dentro l'altro ed è del tutto logico che sia così.

Tutto questo può sembrare oltremodo complicato, mentre un esempio permette di rendersi conto che non solo è semplicissimo, ma è anche molto logico. Si pensi ad un lungo saggio che si ritiene di suddividere nei file seguenti:

1. `premesse.tex` che contiene le informazioni per comporre tutto il materiale iniziale, dall'occhiello fino alla prefazione, inclusi tutti gli indici che si vogliono creare.

2. `introduzione.tex` che contiene un capitolo introduttivo.

3. `statodellarte.tex` che contiene una dettagliata descrizione della situazione pertinente prima della teoria esposta nel saggio.

4. `motivazioni.tex` dove si descrivono le motivazioni che portano allo sviluppo della nuova teoria descritta nel saggio.

5. `sviluppo.tex` dove si discute lo sviluppo della nuova teoria.

6. `risultati.tex` dove si discutono i risultati alla luce della nuova teoria.

7. `confronti.tex` dove si discutono e si confrontano i nuovi risultati con quelli che si potevano ottenere applicando le teorie precedenti. Magari questo file contiene anche le considerazioni conclusive.

8. `appendici.tex` dove si raccolgono tutte le informazioni, dalla bibliografia alle informazioni accessorie allo sviluppo della nuova teoria, fino, eventualmente, agli indici analitici e ai glossari.

Si procederà allora a predisporre un 'master file' `nuovateoria.tex` nel quale si mettono solo il preambolo e i comandi per l'inclusione dei vari file:

```
% file nuovateoria.tex
%
\documentclass[⟨opzioni⟩]{⟨classe⟩}
⟨preambolo che contiene tutti i comandi per i pacchetti da usare, le definizioni di
nuovi comandi, e si conclude con:⟩
%
\includeonly{%
premesse,%
introduzione,%
statodellarte,%
motivazioni,%
sviluppo,%
risultati,%
confronti,%
appendici%
```

```
}
%
% Qui finisce il preambolo e comincia il documento
%
\begin{document}

\include{premesse}
\mainmatter
\include{introduzione}
\include{statodellarte}
\include{motivazioni}
\include{sviluppo}
\include{risultati}
\include{confronti}
\backmatter
\include{appendici}

\end{document}
```

Se si elabora il file `nuovateoria.tex` così come è presentato sopra, LaTeX legge, se ci sono, tutti i file dichiarati come argomento ai vari comandi `\include`. Ma si osservi come è scritta la ⟨*lista di file*⟩ che costituisce l'argomento di `\includeonly`; ogni file della lista è scritto all'inizio di una nuova riga la quale è terminata dal segno di commento al fine di non lasciare spazi spuri nella lista. In questo modo c'è la possibilità di inserire un segno di commento all'inizio delle righe che contengono i nomi dei file che *non* si desidera elaborare.

All'inizio della scrittura dell'intero documento si comincerà presumibilmente con il file `premesse.tex`; si metterà allora un segno di commento davanti al nome di tutti gli altri file; quando poi si passa a scrivere `introduzione.tex`, si toglie il segno di commento davanti al nome di questo file e lo si inserisce, invece, davanti al nome di `premesse.tex`. Via via che si procede, si compone un file alla volta e se ne controlla il risultato attraverso il programma di visualizzazione adeguato al tipo del file di uscita. Ogni volta la compilazione ha luogo su un solo file ed è particolarmente veloce; nello stesso tempo i riferimenti incrociati a elementi contenuti nei capitoli precedenti vengono tutti risolti correttamente. Anche le bozze, se le si volesse stampare, hanno le pagine numerate correttamente. Solo alla fine, a lavoro quasi terminato, si tolgono tutti i segni di commento davanti ai nomi dei file della ⟨*lista di file*⟩, e si esegue la composizione dell'intero testo in un unico file di uscita.

Se lo shell editor lo richiede (e quasi tutti gli shell editor lo richiedono) bisogna solo specificare all'editor il nome del 'master file' affinché, quando si compila e l'editor ha aperto solo uno dei file da includere, esso possa lanciare LaTeX sul 'master file', e non sul file correntemente aperto.

Con gli shell editor che comprendono i commenti di autoconfigurazione (le righe magiche) non è necessario fare nulla di speciale perché le specifiche righe

magiche contente in testa ad ogni file incluso dicono esplicitamente all'editor come si chiama il 'master file' su cui eseguire la compilazione.

Questo tipo di organizzazione del file d'entrata diventa più comodo da usare via via che si acquisisce pratica e se ne apprezzano i vantaggi.

Vale la pena di sottolineare che la divisione in capitoli da includere con il comando `\include` va benissimo con i capitoli. Il comando `\input` serve per gli scopi indicati, ma l'esperienza insegna a servirsene con il massimo profitto in diverse altre circostanze; per esempio lo scrivente ha sviluppato l'abitudine di comporre grandi tabelle in file a se stanti da includere con `\input`. Questo modo di procedere deriva dalla constatazione che le grandi tabelle sono complesse da comporre ed è facile commettere errori. Inserite in file a se stanti, e creato un unico piccolo 'master file' che ordini la lettura solo del file contenente la tabella, ogni tabella viene composta e corretta per suo conto e non c'è pericolo che gravi errori, particolarmente difficili da individuare, fermino la compilazione di un grosso file in un punto che non ha nulla a che vedere con la posizione dell'errore, come, ahimè, talvolta succede.

Dalla versione 0.09 anche i file che compongono questo testo sono gestiti con il meccanismo di `\include` e `\includeonly`. Quest'ultimo, collocato nel preambolo, è formato così:

```
\includeonly{%
preliminari,%
presentazione,%
%
sincrona-asincrona,%
nozioni-tipografia,%
ortografia,%
installazione,%
prime-nozioni,%
classi,%
testi-speciali,%
tabelle,%
figure,%
importazione-figure,%
bibtex,%
indici-glossari,%
presentazioni,%
%
matematica1,%
matematica2,%
microgiustificazione,%
nozioni-tipografia,%
ortografia,%
caratteri,%
classi,%
nuovi-comandi,%
```

```
layout,%
documentazione,%
%
pdfarchiviabile,%
conversione,%
simbologia,%
divisione-sillabe,%
codifiche,%
PageOutput,%
errori,%
sintassi-riepilogo,%
indanalitico%
}
```

e, come si vede, ogni capitolo, numerato o non numerato, e ogni appendice viene elencata nella ⟨*lista di file*⟩ nella speciale maniera descritta sopra. Successivamente il corpo del testo è formato dal blocco:

```
\begin{document}
%
\frontmatter
%
\include{preliminari}
\include{presentazione}
%
\mainmatter
\pagestyle{headings}
%
\include{sincrona-asincrona}
\include{nozioni-tipografia}
\include{ortografia}
\include{installazione}
\include{prime-nozioni}
\include{classi}
\include{testi-speciali}
\include{tabelle}
\include{figure}
\include{importazione-figure}
\include{bibtex}
\include{indici-glossari}
\include{presentazioni}
%
\include{matematica1}
\include{matematica2}
\include{filologia}
\include{presentazioni}
```

```
\include{microgiustificazione}
\include{caratteri}
\include{nuovi-comandi}
\include{layout}
\include{documentazione}
%
\include{pdfarchiviabile}
\include{conversione}
\include{simbologia}
\include{divisione-sillabe}
\include{codifiche}
\include{PageOutput}
\include{errori}
\include{sintassi-riepilogo}

\cleardoublepage
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\backmatter

\include{indanalitico}
%
\end{document}
```

I comandi `\frontmatter` e `\mainmatter` specificano l'inizio dei preliminari e l'inizio del corpo del testo e sono presenti anche se gli eventuali file che dovrebbero venire inclusi di fatto non lo sono perché eliminati con segni di commento dalla lista dei file di `\includeonly`; invece di `\backmatter` si è usata solo la specifica `\appendix` che ordina che da quel punto in poi i capitoli sono da trattare come appendici; non si è usato `\backmatter` perché chi ha predisposto la classe *`book`* ha ritenuto che le appendici facessero parte del corpo del testo e non delle parti finali.

5.9 Gestione degli errori

Purtroppo talvolta i file sorgente contengono errori; in queste circostanze l'interprete TeX procede ad interpretare il flusso di informazioni che via via legge dai file sorgente, finché riesce a dare loro un qualche significato, poi si ferma con un messaggio d'errore. Quasi sempre si tratta di un comando scritto con un errore di ortografia; sullo schermo appare allora il messaggio d'errore con una qualche frase esplicativa e la linea dove TeX si è bloccato scritta 'spezzata' in corrispondenza dell'ultimo oggetto che TeX ha letto. Se si tratta di un errore di ortografia in un comando, appare appunto il comando errato alla fine del primo moncone della riga presentata e la riga del flusso d'entrata non ancora sottoposto al processo di interpretazione appare subito sotto. Il primo moncone di riga è anche preceduto dal numero della riga del file sorgente.

Bisogna osservare che questi arresti ad ogni errore si manifestano solo se lo shell editor ha imposto al compilatore di comporre in 'error stop mode'; TeXShop, per esempio è configurato in questo modo; TeXstudio è invece configurato per comporre in 'scroll mode' (vedi poco più avanti).

Le risposte possibili ad un messaggio d'errore sono quelle di introdurre una delle seguenti lettere: `Invio`, `x`, `q`, `s`, `h`, `i`, `e`, eventualmente seguita da un stringa di testo.

`Invio` Premendo il tasto `Invio` il compilatore ignora l'errore e prosegue la compilazione fino all'errore successivo o fino alla fine del documento, se non incontra altri errori gravissimi. Di solito non è conveniente premere il tasto `Invio`, ma è preferibile premere i tasti corrispondenti alle altre funzionalità del compilatore.

`x` Serve per terminare subito la compilazione; successivamente si può dare il comando allo shell editor di collocare il cursore nella riga contenente l'errore in modo da poterlo correggere e riprendere la compilazione dall'inizio.

`q` Serve per ordinare al compilatore di continuare in modo 'quieto', cioè senza fermarsi ad ogni errore e senza emettere messaggi d'errore. Ovviamente se a un certo punto il compilatore incontra un errore grave che lo blocca permanentemente, esso is ferma presentando come segnale di prompt un asterisco, non più un punto interrogativo; l'asterisco richiede che si immetta qualche cosa che il compilatore riesca a interpretare; non facile da dirsi, anzi questa è una condizione d'errore gravissimo che non ammette recuperi;. Molti shell editor hanno un tasto o un bottone da cliccare che consente di 'abortire' la compilazione in modo pulito.

`s` Serve per ordinare al compilatore di proseguire la compilazione in 'scroll mode', vale a dire proseguendo la compilazione come in 'quiet mode', ma scrivendo sulla console i messaggi d'errore; se si ferma con un asterisco bisogna procedere come descritto sopra.

`h` Serve per chiedere aiuto; i messaggi informativi che LaTeX fornisce sullo schermo talvolta permettono di capire di che errore si tratti; talvolta sono troppo generici e bisogna armarsi di pazienza e di ragionamento investigativo per trovare che cosa c'è che non va.

`e` Serve per uscire dal programma di compilazione e di ritornare immediatamente allo shell editor con il cursore già messo nella riga con l'errore; questa operazione ha esito felice se c'è una perfetta integrazione fra compilatore e shell editor; ma talvolta non riesce perché lo shell editor non è configurato correttamente.

`i` Serve per introdurre una stringa di testo *al posto* dell'ultimo oggetto letto dall'interprete; per esempio se si scrivesse `\iput`, TeX si fermerebbe dopo aver letto questo oggetto a cui non sa dare un significato, perché è un

comando scritto male; al prompt di TeX (costituito da un punto interrogativo) basta scrivere `i\input` e poi premere il tasto di invio; TeX sostituisce `\input` all'oggetto ignoto `\iput` e procede nella compilazione; si noti che il file sorgente non è stato toccato da questa operazione, quindi l'errore esiste ancora e bisogna correggerlo; bisogna cioè ritornare allo shell editor e chiedergli di collocarsi alla riga che contiene l'errore.

Se si ha l'accortezza di lanciare LaTeX abbastanza sovente, ogni possibile errore, anche difficilissimo da trovare, può trovarsi solo nel testo sorgente introdotto fra la penultima e l'ultima volta che si è eseguita la compilazione; se questa viene eseguita sovente, il testo introdotto è relativamente corto e quindi risulta più facile trovare l'errore.

Per errori più complessi bisogna ricorrere a pacchetti oppure a comandi specifici, la cui discussione è fuori luogo in questo capitolo introduttivo destinato alle prime nozioni; Per comprendere meglio certi errori e certi avvisi di cattiva composizione (`Overfull hbox...` e altri avvisi simili), conviene leggere il capitolo 27. Il trattamento degli errori più gravi e gli strumenti disponibili con il sistema TeX sono trattati nel capitolo 28.

Capitolo 6

LaTeX: i vari tipi di documenti e stili di composizione

6.1 Introduzione

I file di classe, che servono per definire i parametri compositivi del particolare documento che si intende comporre, sono il cuore di LaTeX e sono proprio loro che interpretano le istruzioni LaTeX per eseguire la composizione in accordo con lo stile desiderato.

6.2 Classi standard

LaTeX viene distribuito con una piccola dotazione di file di classe abbastanza generici da poter affrontare la composizione di qualunque documento; tuttavia questa genericità ha favorito lo sviluppo di classi particolari di cui si dirà brevemente nei prossimi paragrafi.

book è la classe con cui è stato composto questo testo. Essa prevede di usare di default tutte le strutture di sezionamento, da `\chapter` fino alla più minuta suddivisione del testo costituita da `\subparagraph`; prevede la composizione sul recto e sul verso dei fogli, prevede la pagina contenente il titolo come pagina a sé stante; il disegno grafico prevede uno spostamento della gabbia verso il margine interno; le testatine contengono il titolo corrente e il numero della pagina; il titolo corrente è scritto in caratteri maiuscoli inclinati; il documento può essere suddiviso in materiale iniziale, corpo del testo e materiale finale, con caratteristiche compositive adeguatamente diverse, eccetera.

report è la classe con cui si scrivono i rapporti 'tecnici': testi relativamente brevi, di solito non superiori ad una cinquantina di pagine, con il titolo facente parte della prima pagina dove comincia anche il testo; di default sono composti sul recto delle pagine, per cui la gabbia del testo è centrata. Non prevede la definizione dei comandi `\frontmatter`, `\mainmatter` e `\backmatter` perché un rapporto non è così strutturato come un libro, ma per il resto la classe assomiglia abbastanza alla classe *book*.

article è la classe con cui vengono composti brevi articoli, di solito di lunghezza non superiore alla decina di pagine, con il titolo sulla stessa pagina dove comincia il testo; spesso sono scritti su due colonne; hanno esplicitamente l'ambiente *abstract* per redigere un breve sunto del contenuto; non è definito il comando `\chapter` anche se continua ad essere definito il comando `\part`; nessuno vieta di dividere un articolo in parti, ma la struttura di sezionamento più alta è `\section`; spesso gli articoli sono senza indice generale, né indice analitico, ma dispongono quasi sempre di una bibliografia.

letter serve per comporre lettere commerciali; lo stile è molto americano, ma non è così difficile personalizzarlo per conformarsi allo stile italiano delle lettere commerciali; consente, volendo, di scrivere lettere con il medesimo testo ad una moltitudine di destinatari, così come consente di scrivere diverse lettere a destinatari diversi inserendole nello stesso file sorgente.

ltnews serve per comporre una semplice newsletter di solito di una sola pagina, composta su due colonne.

ltxdoc serve per comporre la documentazione dei file sorgente dei pacchetti che vengono distribuiti; questi file sorgente contengono simultaneamente la documentazione e il codice e possono servire per produrre diverse classi e/o file di estensione; il loro formato è molto particolare, ma sono preziosi per i programmatori per poter scrivere dei programmi in linguaggio TEX o LATEX in una forma ben documentata, senza che questi commenti disturbino il lavoro del motore TEX quando deve comporre un documento.

ltxguide serve per comporre le guide di documentazione dei vari aspetti di LATEX; si tratta delle guide incluse in ogni distribuzione e si trovano generalmente nella cartella `.../doc/latex/base/`; queste guide sono la fonte principale di informazione che l'utente dovrebbe consultare e 'conoscere a memoria'...

minimal è una classe minima, fatta apposta per avere una classe che garantisca il minimo di performance, essenzialmente pensata per collaudare la definizione di nuovi comandi o lo sviluppo e le prestazioni di nuovi pacchetti di estensione quando il programmatore li sta costruendo.

proc è una classe simile ad *article* che serve per produrre degli articoli secondo uno stile da *proceedings* (atti di una conferenza).

slides serve per predisporre dei lucidi di presentazione di un discorso o di una conferenza. In realtà una volta si usavano le lavagne luminose; oggi si usano i videoproiettori; con questi ultimi non si producono lucidi ma si proietta direttamente dal calcolatore; le prestazioni di *slides* sono adatte proprio ai lucidi su fogli di metacrilato. Per le presentazioni oggi si possono usare diversi programmi, ma, restando in ambito LaTeX, esistono delle classi, come *beamer*, ottenuta con il pacchetto *beamer*, che producono delle presentazioni a colori bellissime, con animazioni finalizzate alla presentazione, e con la qualità della composizione della matematica e della gestione/importazione di immagini efficaci come quelle che solo LaTeX può offrire.

6.3 La creazione di nuove classi

L'utente può crearsi nuove classi con due approcci (in verità non mutuamente esclusivi) molto semplici; (*a*) modificare le prestazioni di una classe esistente e (*b*) creare una classe ex novo.

Il primo approccio è decisamente il più semplice ed è affrontabile da chiunque; il file `clsguide` che si trova nella solita cartella `.../doc/latex/base/` contiene le indicazioni per seguire questa via e presenta non pochi esempi.

Il secondo approccio è decisamente molto più complesso, ma consente una personalizzazione maggiore. A questo proposito può essere interessante leggere la documentazione relativa alla classe *octavo* in `.../doc/latex/octavo/octavo.pdf`. Questa classe è un rifacimento della classe *book* dove però sono state fatte scelte particolari sia nel dimensionamento della gabbia e delle altre informazioni 'geometriche' per la composizione, ma sono stati anche ridefiniti numerosi comandi per ottenere risultati particolari; usando la classe *octavo* al posto della classe *book*, il compositore non deve imparare nessun comando nuovo; al massimo deve familiarizzarsi con le nuove opzioni per il formato della carta che si riferiscono alla terminologia tradizionale britannica. Se nel preambolo di un documento da comporre con la classe *book* si sostituisce la parola *book* con la parola *octavo*, la composizione risulta assai simile, salvo la giustezza e la densità delle righe che risultano perfettamente adattate al corpo dei caratteri usati; ma se si specifica qualche opzione scelta fra quelle nuove, l'aspetto della composizione cambia in modo abbastanza significativo. La documentazione è molto istruttiva e spiega sia i motivi delle scelte fatte sia come queste siano state rese nel linguaggio di programmazione di TeX; il compositore che vuole cimentarsi nella creazione di nuove classi ha molto da imparare dalla lettura di questa documentazione.

Una via di mezzo è quella di utilizzare classi non standard ma fortemente personalizzabili come si vedrà nel prossimo paragrafo.

Infine, la via maestra è quella di non creare nessuna classe nuova, a meno che non sia fortemente indispensabile. Esistono infiniti pacchetti di estensione

per modificare le prestazioni delle classi esistenti; qui se ne citeranno solamente alcuni, quelli più usati e considerati più utili.

fancyhdr serve per personalizzare le testatine ed i piedini; di ognuno si può personalizzare quello che compare nella pagina di sinistra, oppure nella pagina di destra; di ognuno si può specificare che cosa scrivere nel centro, a sinistra, oppure a destra, così da avere sei posizioni in ogni pagina per inserire informazioni che, in definitiva, aiutano a navigare nel testo, e quindi sono particolarmente indicate nei documenti molto strutturati. Mentre le testatine 'ordinarie', per esempio della classe *book*, oltre al numero della pagina, possono contenere solo il titolo (corto) del capitolo o del paragrafo correnti, con *fancyhdr* si riesce anche a inserire informazioni di terzo livello, per esempio il titolo dei sottoparagrafi.

caption serve per personalizzare la composizione delle didascalie.

geometry serve per personalizzare il layout della pagina, la sua geometria, secondo schemi tradizionali o moderni, sempre parametrizzati, in modo da essere completamente liberi nella definizione dei margini, dei contrografismi fra testatina o piedino e il corpo della pagina; per scegliere la giustezza in modo da avere il numero (medio) ottimale di caratteri in ogni riga in relazione al font usato e al suo corpo 'normale'. Sotto molti aspetti *geometry* agisce come il pacchetto *typearea*, apparentemente molto popolare fra gli esperti che scrivono sul forum GUIT, ma è decisamente più agile e consente di personalizzare in modo più esteso.

Ovviamente questi pacchetti sono utili anche nel creare nuove classi, perché basta caricarli all'inizio della nuova classe, specificando loro i parametri desiderati, così che spesso la creazione di una nuova classe si riduce veramente a poco.

6.4 Alcune classi non standard

Il sistema TEX viene distribuito con una enorme quantità di file aggiuntivi, compresi i file di classe che alcuni autori hanno predisposto e messo a disposizione degli utenti.

6.4.1 Le classi Komascript

Markus Kohm (da cui il prefisso 'Koma') ha messo a disposizione le quattro classi fondamentali per la composizione di libri, rapporti, articoli e lettere, rispettivamente *scrbook*, *scrreprt*, *scrartcl* e *scrlttr2*. Le caratteristiche e i comandi disponibili sono reperibili nella documentazione del pacchetto *koma-script*.

Ma la componente di questo pacchetto certamente più utile è il pacchetto *typearea*, richiamabile esplicitamente dai file di classe, che permette di disegnarsi a proprio piacimento la griglia di scrittura secondo i criteri più diffusi; il pacchetto

typearea è richiamabile anche da altri file di classe, non è una esclusiva della collezione Komascript di Markus Kohm.

In questa collezione di classi, la classe *scrlttr2* è molto interessante perché descrive il layout delle lettere in una maniera più adatta alle tradizioni europee; una caratteristica particolare è la sua capacità di adattare il formato della carta e quello della busta, persino delle buste con finestra trasparente, molto usate nella corrispondenza burocratica e commerciale. La documentazione spiega anche come predisporre nuove opzioni e/o configurazioni locali adatte a formati di carte e di buste non conformi alla norme ISO, anche nella versione molto dettagliata delle norme ISO adattate al Giappone. Per quel che riguarda l'Italia, esistono delle norme relative all'abbinamento della carta con la busta, ma apparentemente non esistono delle norme per le buste con finestra.

Le classi della collezione Komascript sono alla base anche di classi o di pacchetti molto in voga fra i frequentatori del forum di GuIT, in particolare per il pacchetto *ClassicThesis*. Questo fatto discende anche dalla facilità con cui è possibile configurare il layout della pagina e gli stili compositivi di tutte le parti di sezionamento, dai titoli correnti di ogni livello, alle didascalie, dalle testatine ai piedini.

6.4.2 La classe *memoir*

Peter Wilson ha scritto questa unica classe con la quale è possibile comporre quasi tutto quello che si può comporre con le classi standard, ma offre possibilità di personalizzazione difficili da realizzare se ci si appoggia alle classi standard.

Una parte pregevole di questo lavoro è costituita dal file contenente il manuale di documentazione, assai lungo e dettagliato, contenente anche una buona dose di consigli stilistici e una discreta storia del disegno grafico del libro e del documento[1]; per le parti tecniche il manuale è essenziale perché l'estensione di LaTeX e delle sue classi che la classe *memoir* offre è decisamente importante.

La classe permette di personalizzare qualunque aspetto del documento, dal disegno grafico della pagina, alla scelta dei font per scrivere qualunque testo speciale, compresi i titolini delle sezioni, le testatine e i piedini. Qualunque cosa possa essere personalizzata, e che è difficile personalizzare usando LaTeX standard, con *memoir* può essere fatta facilmente e con poche istruzioni.

[1] Questo, in verità, appariva sul manuale di *memoir* fino a qualche anno fa; ora il manuale è molto più tecnico e chiarisce ogni possibile dubbio di programmazione, ma in verità quell'excursus storico era molto interessante. L'excursus storico è ancora disponibile come file a parte ed è leggibile con il comanda da terminale `texdoc memdesign`. L'autore, Peter Wilson scrive:
These notes briefly cover some aspects of book design and typography, independently of the means of typesetting. Among the several books on the subject listed in the Bibliogra- phy I prefer Bringhurst?s *The Elements of Typographic Style* (Bringhurst 1999).
The notes originally formed the first part of a user manual for the memoir class for use with the LaTeX typesetting system developed by Leslie Lamport (Lamport 1994) based on Donald Knuth?s TeX system (Knuth 1984). The manual was first published in 2001 and as the notes have grown in size and memoir?s capabilities have been extended the manual also grew to approaching 700 pages (Wilson 2009). At that point seemed advantageous to separate the design notes from the technicalities, hence this document.

L'utente che si rivolge a *memoir* finisce con restarci affezionato per sempre; ma deve fare molta attenzione con le personalizzazioni, perché queste sono utili, certamente, ma come si può migliorare qualunque cosa, la si può anche peggiorare. In altre parole questa classe è decisamente più adatta ad un utente esperto, cosa che riesce a chiunque abbia un minimo di sensibilità estetica ed abbia spirito di osservazione per imparare e valutare i pregi e i difetti di ciò che viene fatto da altri. Dagli altri si dovrebbe prima di tutto imparare ad evitare gli errori, cercando di imitare il meglio e di avvicinarsi il più possibile alle realizzazioni degli artisti grafici più quotati; con lo strumento *memoir* ci si può riuscire.

L'autore di *memoir* nel suo documento `memdesign` racconta cose che hanno a che vedere con la cultura tipografica oltre che con il book design. Vale la pena di citare questo brano tradotto in italiano da Luciano Battaia.

> L'essenza di un libro ben stampato è che non si fa notare al primo, o addirittura al secondo o successivo, sguardo di chiunque non abbia un occhio allenato. Se la vostra prima reazione nello sfogliare un libro è di fare un'esclamazione di meraviglia osservando il layout, allora il libro è molto probabilmente mal progettato, se mai è stato progettato. La stampa di qualità è raffinata, non stridente.
>
> Con l'avvento del desktop publishing molti autori hanno la tentazione di progettare da soli i loro testi. Sembra molto facile farlo. Basta scegliere alcune delle migliaia di font disponibili, usarne uno per i titoli, uno per il testo principale, un altro per le didascalie, decidere le dimensioni dei caratteri, e la cosa è fatta.
>
> Tuttavia, come scrivere è un'abilità che bisogna apprendere, anche comporre tipograficamente un testo è un'arte che si deve apprendere e su cui bisogna esercitarsi. Ci sono centinaia di anni di esperienza racchiusi nel buon design di un libro. Essi non possono essere trascurati con leggerezza e molti autori che progettano i loro libri non conoscono alcune delle conquiste più importanti, per non parlare del fatto che quello che fanno è esattamente in antitesi con esse. Un esperto può infrangere le regole, ma allora sa che ha delle buone ragioni per farlo.
>
> [...] Se un libro grida 'guardami', questo è un segnale, e un pessimo segnale, per chi l'ha progettato.

6.4.3 La classe *ncc*

Alexander Rozhenko ha predisposto una classe *ncc*, valida per tutte le lingue accessibili con *babel*, ma pensata inizialmente per il russo, che permette di comporre articoli, rapporti e libri semplicemente specificando opportune opzioni a questa unica classe. La classe è accompagnata da un certo numero di file di estensione che consentono di eseguire composizioni particolari; l'autore non le ha considerate di uso generale, cosicché ha potuto mantenere le dimensioni della classe entro limiti ragionevoli, ma ha consentito di raggiungere prestazioni

supplementari con moduli aggiuntivi da usarsi di volta in volta a seconda delle necessità compositive.

La classe è estremamente versatile e consente di unire le prestazioni di molte estensioni LaTeX in un'unica classe. Cito a caso gli allineamenti di equazioni, quelli di *amsmath* tanto per intenderci, realizzati con pochi comandi assai versatili; gli oggetti flottanti 'mini', per minifigure e minitabelle, comprese le (piccole) figure e tabelle a margine del testo; vari tipi di teoremi già predisposti, senza bisogno di ricorrere a `\newtheorem`; ovviamente ci sono anche i comandi per le dimostrazioni.

Tutti i comandi dispongono di parametri facoltativi che consentono una personalizzazione piuttosto ampia.

La documentazione è abbastanza dettagliata e si trova in `.../doc/latex/ncclatex/ncclatex.pdf`; è opportuno studiare bene la documentazione, perché, sebbene si tratti sempre di mark-up LaTeX, esistono moltissimi nuovi comandi e molti ambienti sono sostituiti da comandi con diversi argomenti in parte facoltativi e in parte obbligatori. Bisogna, insomma, prenderci la mano, ma ne vale la pena.

6.4.4 Le classi per le tesi di laurea

Spesso gli studenti si avvicinano a LaTeX quando devono scrivere la tesi di laurea. Dagli archivi CTAN si possono scaricare diversi pacchetti che contengono il necessario per comporre la tesi di laurea o di dottorato. Fra i tanti file di estensione, che servono par estendere alla composizione delle tesi ad alcune classi, e le varie classi si citano i pacchetti *TesiClassica* [47], *TesiModerna* [48], *ClassicThesis* [38], *TOPtesi* [5], *sapthesis* [7], *suftesi* [65], *frontespizio* [26], per lo più scritte da italiani per gli studenti universitari italiani.

ClassicThesis scritta da un docente tedesco, ma adatta a tutte le lingue, offre un bellissimo design della pagina, che sarebbe quanto mai indesiderabile personalizzare, perché si perderebbe tutto il bello di questo pacchetto. Siccome il layout della pagina è piuttosto originale, può non adattarsi alle specifiche di questa o quella università.

Anche *TesiClassica* e *TesiModerna*, che sono figlie di *ClassicThesis* ma più sobrie, potrebbero essere difficili da personalizzare. I layout di pagina, leggermente diversi, sono molto belli.

sapthesis e *suftesi* permettono diverse opzioni per i layout della pagina, ma eseguita questa scelta sarebbe meglio non metterci più le mani per personalizzare ulteriormente. Va notato che fra le opzioni di *suftesi* ce ne sono diverse per produrre articoli e libri in formati diversi dal solito formato di carta A4; uno di questi formati e stili di composizione imita molto bene quello usato da Robert Bringhurst per comporre il suo testo *The Elements of Typographc Style* [8]. *suftesi* è fortemente personalizzabile, ma non nel disegno della pagina, per cui potrebbe essere incompatibile con le specifiche richieste da alcune università.

Il pacchetto *frontespizio* di Enrico Gregorio si dedica esclusivamente al frontespizio della tesi; questo è completamente configurabile in ogni suo dettaglio,

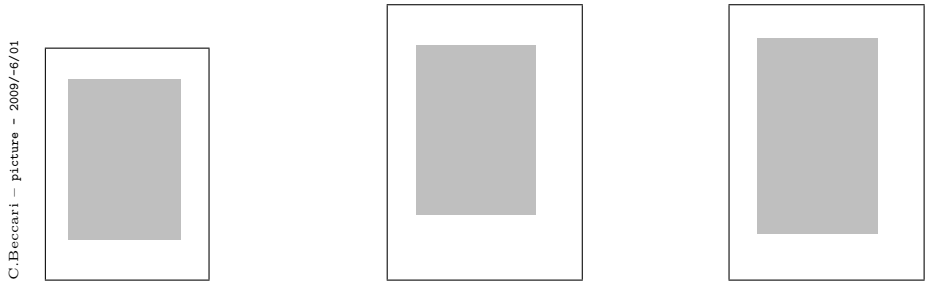
per cui è possibile predisporre il frontespizio della tesi virtualmente per ogni prescrizione di segreteria di facoltà e/o per ogni ateneo. L'unica cosa a cui bisogna fare attenzione è che questo bellissimo pacchetto *non* consente di fare le 'porcherie' richieste in alcune facoltà; evidentemente chi ha redatto quelle prescrizioni o non è ancora passato al calcolatore e usa ancora la macchina da scrivere, o usa solamente e male un word processor (che usato bene produrrebbe anche risultati buoni), oppure ignora completamente i rudimenti della tipografia. Gli studenti universitari che si accingono a scrivere la tesi non si scoraggino: se una cosa non si può fare con *frontespizio* allora vuol dire che è meglio non farla!

Il pacchetto *TOPtesi* contiene sia il file di classe, sia un pacchetto omonimo con cui si può configurare un certo numero di classi standard, contiene un pacchetto con comandi utili, che può essere usato indipendentemente dalla classe, e un pacchetto per il frontespizio, con il quale probabilmente si possono personalizzare diverse classi, ma non è stato creato come modulo a parte espressamente per questo scopo.

La classe è predisposta per comporre tesi in italiano e in inglese (anche il frontespizio): per comporre il frontespizio in lingue diverse dalle due citate, si possono usare comandi specifici che possono essere inseriti in un file di configurazione ed è stata usata per scrivere tesi in francese, spagnolo, tedesco, visto che uno degli obbiettivi dell'autore era quello di mettere a disposizione un pacchetto in grado di soddisfare le esigenze degli studenti in mobilità Erasmus; perciò il pacchetto è personalizzabile per ogni lingua e per molti stili universitari. La classe è stata pensata anche per scrivere le tesi completamente in lingue diverse dall'italiano e dall'inglese in vista del fatto che gli studenti in doppia laurea con i programmi Erasmus devono scrivere la tesi anche (o solo) nella lingua dell'università ospitante.

Questa classe è nata come una sovrastruttura della classe *report* ma, usando solo il file di estensione, non la classe, funziona anche con la classe *book*; la classe *toptesi*, benché costruita sopra la classe *report*, che non distingue le tre parti di un documento complesso, invece, fa questa distinzione automaticamente: essa comincia a comporre con lo stile della `\frontmatter` come nella classe *book*, ma nel momento in cui viene eseguito il suo comando `\indici`, vengono composti i vari indici richiesti e poi viene eseguito l'equivalente del comando `\mainmatter`; con il comando `\appendix`, di fatto si passa automaticamente alla composizione del materiale finale.

Nella classe *toptesi* le appendici sono forzatamente nella parte finale, ma vengono identificate con "numeri" costituiti dalle lettere maiuscole. Invece nella *book*, il comando `\backmatter` elimina la dicitura 'Appendice ⟨*lettera*⟩' prima del titolo di ogni appendice e le compone con capitoli non numerati. La classe *toptesi*, lo si ripete, prevede che le appendici facciano parte del materiale finale, e le tratta come normali capitoli numerati (con lettere maiuscole). Ovviamente la questione se le appendici facciano o non facciano parte del materiale finale è una questione di gusti e di tradizione tipografica; il grande e citatissimo 'book designer' Bringhurst attribuisce le appendici al materiale finale. Chi ha



(*a*) (*b*) (*c*)

Figura 6.1: Tre geometrie di pagina a confronto. (*a*) Gabbia (in grigio) e margini per questo testo composto con la classe `book` su foglio in formato B5; (*b*) gabbia e margini in rapporto aureo fra di loro su su foglio in formato A4; (*c*) gabbia e margini ottenuti con l'estensione `layaureo` su foglio in formato A4.

predisposto la classe `book` attribuisce, invece, le appendici al materiale centrale del documento.

Molti dei pacchetti citati sono usabili anche con XeLaTeX; questo è particolarmente utile per tesi che trattino di lingue scritte in alfabeti diversi da quello latino o anche quando contengano una simbologia molto ricca.

6.4.5 L'estensione `layaureo`

Fabiano Busdraghi ha prodotto un pacchetto di estensione che non modifica gli altri parametri della classe in uso, ma imposta il disegno grafico della pagina secondo il criterio della sezione aurea; il pacchetto si chiama `layaureo` e nel modo più semplice possibile, senza ricorrere ad impostazioni di parametri o di parole chiave talvolta 'immaginose', modifica la larghezza del testo e la sua altezza, nonché i margini di pagina in modo da ottenere il disegno grafico basato sulla sezione aurea.

In particolare allarga la giustezza in modo da coprire meglio la pagina A4 di quanto non lo facciano le classi standard; questo allargamento dipende dal corpo normale del font in uso. Poi se il documento deve essere composto sul recto e sul verso delle pagine, allora sistema il blocco del testo in modo che i margini stiano fra loro come la sezione aurea. È possibile usare una opzione per spostare tutto il testo di una quantità fissa verso l'esterno, per esempio 5 mm, al fine di tenere conto della rilegatura.

Si ricorda che la sezione aurea è un rapporto usato molto spesso nelle arti grafiche e lo si ritrova in moltissime strutture naturali; per la sua definizione si veda in particolare l'equazione etichettata (sezione aurea) nella pagina 252.

Senza tenere conto dell'aumento eventuale del margine interno per tenere conto della rilegatura, la scelta della sezione aurea implica, per esempio, che il margine interno sia la frazione 0,618... del margine esterno.

Vale la pena di mettere a confronto tre layout di pagina nella figura 6.1; il primo layout corrisponde esattamente a questo testo composto con la classe `book` alla quale non sono state apportate modifiche geometriche di nessun genere.

Il secondo layout ha la pagina, il testo e i margini in rapporto aureo fra di loro; per essere più precisi, il rettangolo della pagina rappresenta in scala il foglio UNI A4, i cui lati hanno un rapporto di $\sqrt{2}$ fra di loro e non è quindi un rettangolo aureo; però la larghezza della pagina e la giustezza del testo sono in rapporto aureo, così come lo sono i margini esterno ed interno fra di loro; gli stessi rapporti valgono per le dimensioni verticali; a causa di questa regolarità delle proporzioni si può osservare che la diagonale NW–SE della gabbia del testo coincide con la diagonale della pagina. Si noti anche che la gabbia del testo è un rettangolo UNI, cioè con il rapporto $\sqrt{2}$ fra l'altezza e la base; esso non ha le proporzioni auree.

Il terzo layout è quello che si ottiene con l'estensione `layaureo` sulla pagina UNI A4 per la quale è stato disegnato; la giustezza è fissata circa 10 mm più grande della giustezza di default della classe `book` ma non è in rapporto aureo con la larghezza della pagina; invece i margini interno ed esterno lo sono fra di loro; i margini superiore e inferiore, al contrario, sono in rapporto di $\sqrt{2}$ e il margine inferiore è il più grande. L'altezza della gabbia è in rapporto aureo con la sua base e forma un rettangolo aureo.

A causa dei margini ristretti, in particolare quello esterno, i layout basati in tutto o in parte sulla sezione aurea non sono adatti ad accogliere le note marginali. La classe `book` usata con una pagina in formato A4, invece che B5 come in questo testo, ha margini esterni abbastanza generosi da poter accogliere anche le note marginali.

La figura 6.2 mostra chiaramente la differenza fra il rettangolo UNI, basato sulla geometria del quadrato, e il rettangolo aureo, basato sulla geometria del pentagono regolare; in entrambi i casi l'altezza del rettangolo è pari alla corda maggiore del poligono regolare avente la base uguale a quella del rettangolo, ma, a pari base, il rettangolo aureo, figura 6.2(b), è significativamente più alto del rettangolo UNI, figura 6.2(a).

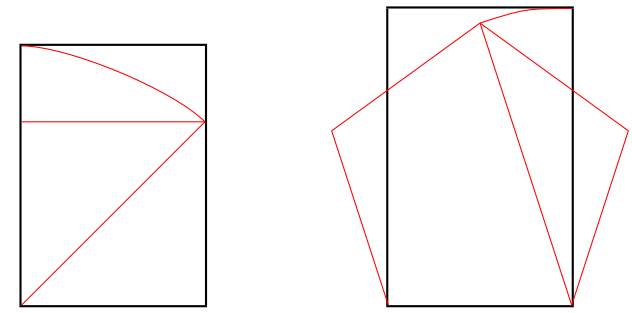
Nella pagina 124 e nella successiva sono mostrate in dettaglio tutte le misure che riguardano le pagine pari e dispari di questo testo. Le misure indicate in punti tipografici sono state ricalcolate da LaTeX stesso, caso mai fossero state modificate in qualche modo; in realtà non è stato modificato nulla rispetto alle impostazioni di default.

6.5 I pacchetti di estensione

Anche nel paragrafo precedente si è parlato di alcuni pacchetti di estensione; virtualmente se ne parla in ogni capitolo.

I pacchetti di estensione, spesso chiamati *file di stile* a causa dell'estensione `.sty` che essi hanno, estendono le capacità di LaTeX mettendo a disposizione del compositore nuovi comandi o nuovi stili di composizione.

C. Beccari – picture – 2009/06/01



(*a*) (*b*)

Figura 6.2: Costruzioni geometriche per mettere in relazione l'altezza del rettangolo con il lato del poligono regolare sul quale è costruito

6.5.1 Come invocare i file di estensione

Nel preambolo di questo testo sono stati invocati diversi pacchetti di estensione; di alcuni si è già detto nel capitolo 5. Mentre i pacchetti descritti in quel capitolo sono da usarsi virtualmente ogni volta che si scrive un documento scritto con una tastiera nazionale (probabilmente in Italia con una tastiera italiana), con un alfabeto latino che contiene lettere accentate e in una lingua diversa dall'inglese, ora si mostrano le invocazioni degli altri pacchetti richiesti per la composizione di questo specifico testo:

```
\usepackage{pict2e}[2004/08/01]
\usepackage{graphicx}
\usepackage{mflogo}
\usepackage{amsmath,amssymb}
\usepackage{afterpage}
```

Il comando di invocazione è `\usepackage` che segue la sintassi seguente:

`\usepackage[`⟨*opzioni*⟩`]{`⟨*pacchetto*⟩`}[`⟨*data*⟩`]`

Le ⟨*opzioni*⟩ sono quelle specifiche del pacchetto invocato; per esempio per `babel` il pacchetto era stato invocato nel preambolo con il comando

```
\usepackage[italian]{babel}
```

Il ⟨*pacchetto*⟩ può consistere nel nome di un solo file senza la specificazione dell'estensione `.sty`, ma può anche consistere in una lista di nomi separati l'uno dall'altro da una virgola. La ⟨*data*⟩ è la data scritta secondo le norme ISO nella forma *aaaa/mm/gg* (anno con tutte quattro le cifre, mese indicato con due cifre anche se la prima è uno zero, giorno indicato con due cifre anche se la prima è uno zero) che indica la data rispetto alla quale quella del pacchetto deve essere uguale o posteriore; questa indicazione serve per essere sicuri di

usare pacchetti sufficientemente aggiornati; come si vede nell'esempio riportato sopra, si è richiesto il pacchetto *pict2e* di data successiva o uguale al 1° agosto 2004, perché le versioni precedenti a questa data non disponevano di certi nuovi comandi introdotti in un secondo tempo.

Si fa notare che le ⟨*opzioni*⟩ indicate nell'invocazione del pacchetto sono *locali* a quel pacchetto e se questo non le riconosce viene emesso un avvertimento che informa che l'opzione tale-e-tale non è stata riconosciuta. Invece le ⟨*opzioni*⟩ specificate nella invocazione del file di classe attraverso il comando `\documentclass` sono *globali*, nel senso che esse vengono passate anche alle successive invocazioni dei pacchetti; l'opzione *italian* specificata per *babel*, invece, avrebbe potuto essere indicata fra le opzioni di `\documentclass` e così avrebbe potuto essere usata, senza bisogno di ripeterla, anche da altri pacchetti, come per esempio *varioref* o *layout*; il primo esegue le citazioni nella forma, per esempio, '... nella figura "3.5 della pagina 123"...'; il secondo pacchetto serve per disegnare in una pagina a se stante il layout grafico della pagina, indicando graficamente i rettangoli che descrivono la gabbia, o la testatina, e a parole le indicazioni metriche che nel disegno sono riportate solo mediante delle frecce. È appunto il pacchetto che è stato usato per disegnare i layout della pagina 124 e della successiva.

6.5.2 I vari pacchetti e gli archivi internazionali

È facile comprendere che i pacchetti disponibili siano numerosissimi; l'archivio internazionale CTAN (che sta per Comprehensive TEX Archive Network) ne contiene diverse migliaia ottenuti come contributi dei vari utenti di LATEX che, come succede con il software libero, hanno realizzato qualche estensione per risolvere qualche loro problema contingente e hanno ritenuto che fosse utile anche per gli altri utenti; perciò hanno caricato il pacchetto nell'archivio in modo che chiunque potesse servirsene liberamente e gratuitamente.

Il sistema degli archivi CTAN è formato da tre archivi principali, uno negli USA, uno nel Regno Unito e uno in Germania, che sono sempre sincronizzati l'uno con l'altro e costituiscono la sorgente principale delle estensioni e delle distribuzioni gratuite del sistema TEX. Sparsi per il mondo ci sono poi innumerevoli siti che costituiscono i *mirror* dei tre archivi principali; questi mirror vengono sincronizzati dai loro amministratori quasi in tempo reale, ma generalmente non sono in ritardo di più di una settimana rispetto ai tre archivi principali. Per questo motivo c'è sempre la possibilità di avere un archivio abbastanza vicino così che le connessioni internet siano le più rapide possibile.

Tutti gli archivi e quasi tutti i mirror hanno una pagina interattiva che permette di cercare o per nome o per argomento i file che interessano; solitamente sono anche predisposti per permettere di scaricare intere cartelle in formato compresso così da essere sicuri di non dimenticare nulla per la strada.

Le distribuzioni migliori, come MiKTEX per le macchine Windows, dispongono di un wizard, altre di un assistant, altre di un package manager, che permettono

di avere preliminarmente il nome di tutti i pacchetti disponibili e di scaricare sul proprio calcolatore solo i pacchetti che interessano.

Gli archivi principali sono:

```
cam.ctan.org
dante.ctan.org
tug.ctan.org
```

In Italia ci sono due mirror registrati:

```
ctan.mirror.garr.it
cis.uniRoma2.it
```

ma, lingua permettendo, si può accedere ai mirror francesi, svizzeri, austriaci e, in particolare all'archivio dell'associazione Dante (`http://www.dante.de`), localizzato nella Germania meridionale; il mirror localizzato in Svizzera all'indirizzo `ftp://mirror.switch.ch/mirror/tex`, o quello localizzato in Austria all'indirizzo `http://gd.tuwien.ac.at/publishing/tex/CTAN/` sono particolarmente efficaci. Nel seguito con CTAN si indicherà indifferentemente l'insieme degli archivi internazionali o semplicemente l'indirizzo di base di uno degli archivi principali o di un mirror.

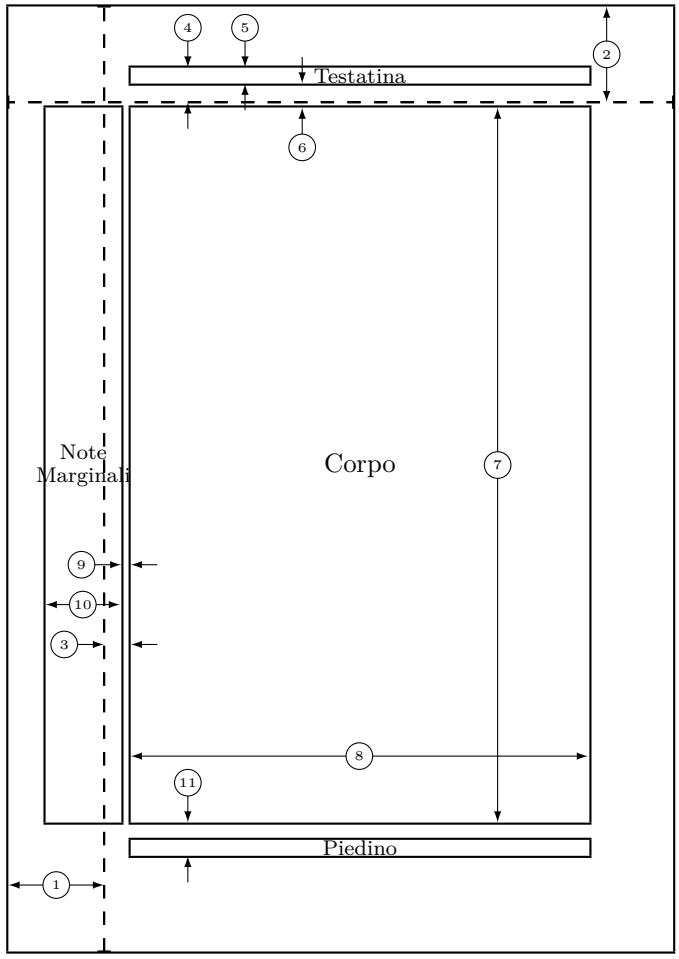
È bene dirlo subito: normalmente non è necessario collegarsi a nessun archivio principale o 'specchiato'; è invece importante sapere come configurare il proprio package manager, o wizard, o assistant per potersi collegare quando è necessario aggiornare la propria installazione completa; questa è una operazione di configurazione da farsi una volta per sempre e poi non ci si pensa più; ci si collega senza dover specificare nessun indirizzo internet, ci si aggiorna, e il gioco è fatto. È più importante collegarsi al sito ufficiale `www.tug.org` dell'associazione internazionale degli utenti di TEX (TEX Users Group). dove si possono trovare notizie interessanti, situazione di aggiornamento degli archivi, documentazione, DVD, notizie sulle conferenze internazionali e nazionali.

6.6 Come scrivere nuovi pacchetti

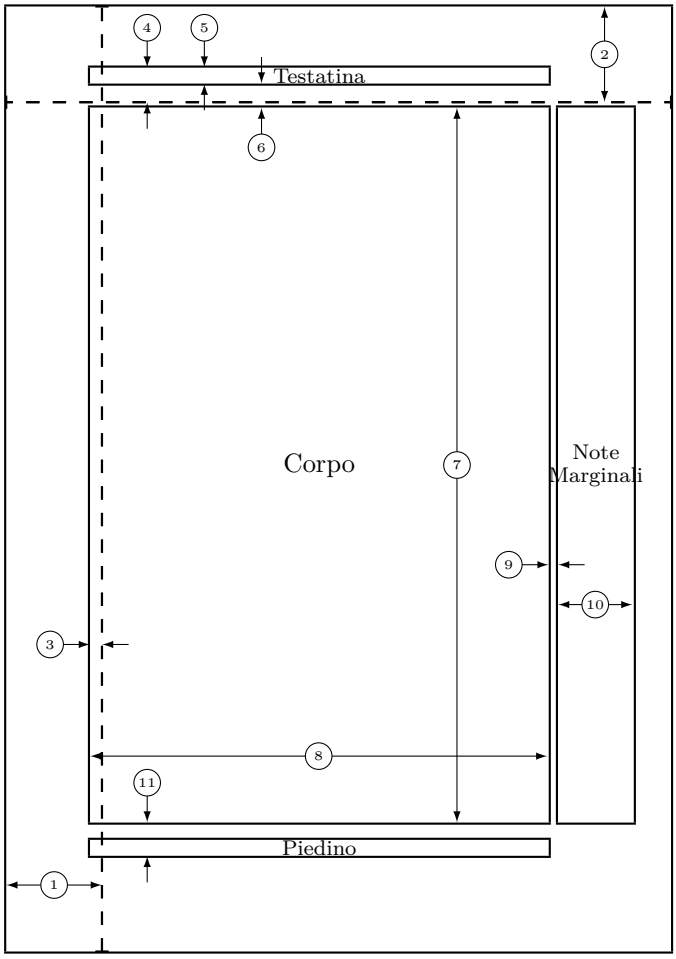
Cercando bene si può quasi sempre trovare il problema già risolto in un qualche pacchetto presente negli archivi.

Tuttavia talvolta può essere necessario crearsi un pacchetto di estensione personale; per questo scopo bisogna leggere con attenzione la già citata guida `clsguide` nella cartella del proprio sistema dedicata alla documentazione `.../doc/latex/base/`; si scoprirà che la cosa di per sé è facilissima; se c'è qualche difficoltà questa risiede nella scelta di che cosa mettere nel pacchetto personale.

C'è però una risposta semplicissima; tutti hanno bisogno di estensioni personali e prima o poi si scrivono dei nuovi comandi ad uso personale, che rispecchiano il proprio stile di gestione del file sorgente. Possono essere macro per inserire certi loghi o certi marchi che hanno a che fare con l'istituzione alla quale si appartiene; possono essere macro per scrivere sigle o abbreviazioni frequenti nei



1 un pollice + \hoffset
2 un pollice + \voffset
3 \evensidemargin = 20pt
4 \topmargin = -26pt
5 \headheight = 12pt
6 \headsep = 18pt
7 \textheight = 538pt
8 \textwidth = 345pt
9 \marginparsep = 7pt
10 \marginparwidth = 57pt
\marginparpush = 5pt (non mostrato)
11 \footskip = 25pt
\hoffset = 0pt
\voffset = 0pt
\paperwidth = 500pt
\paperheight = 711pt



1	un pollice + \hoffset	2	un pollice + \voffset
3	\oddsidemargin = -9pt	4	\topmargin = -26pt
5	\headheight = 12pt	6	\headsep = 18pt
7	\textheight = 538pt	8	\textwidth = 345pt
9	\marginparsep = 7pt	10	\marginparwidth = 57pt
11	\footskip = 25pt		\marginparpush = 5pt (non mostrato)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 500pt		\paperheight = 711pt

testi che si scrivono abitualmente; possono essere nuovi ambienti che riflettono le proprie necessità compositive; possono essere anche semplici modifiche di comandi standard.

Bene, tutto ciò può essere collocato in un file chiamato in una maniera un po' personalizzata, diversamente dal solito `myfile.sty`, per esempio `estensionedi`⟨*iniziali*⟩`.sty`, dove ⟨*iniziali*⟩ può consistere nelle iniziali del proprio nome e cognome o quant'altro personalizzi il vostro file. L'estensione del nome del file *deve* essere `.sty`.

Seguendo le istruzioni della Guida, il file deve cominciare con due righe importantissime:

```
\NeedsTeXFormat{⟨formato⟩}[⟨data⟩]
\ProvidesPackage{⟨nome-del-pacchetto⟩}[⟨data, versione e descrizione⟩]
```

Il ⟨*formato*⟩ se usate LaTeX 2$_\varepsilon$, come si suppone, sarà `LaTeX2e`; la ⟨*data*⟩ sarà la data del formato che voi avete usato per scrivere le vostre macro; l'indicazione di questa data serve per assicuravi che il vostro file per sbaglio non possa venire usato da voi (su un'altra macchina) o da altri con una versione obsoleta del ⟨*formato*⟩; la data, scritta nella forma ISO, cioè *aaaa/mm/gg*, può essere indicata opzionalmente anche nella riga nella quale si richiama con `\usepackage` un pacchetto qualsiasi, in modo da essere sicuri di non usare versioni obsolete; naturalmente dovete conoscere la data del pacchetto non obsoleto che vi serve.

Il ⟨*nome-del-pacchetto*⟩ sarà verosimilmente il nome del vostro file personale, mentre la ⟨*data, versione e descrizione*⟩ deve essere una stringa di parole, possibilmente di lunghezza non maggiore di una riga di caratteri della finestra comandi, o del terminal, o della console; queste informazioni devono avere la forma seguente:

```
⟨aaaa/mm/gg⟩ v.⟨versione⟩ ⟨breve descrizione⟩
```

La data, in formato ISO, è quella della creazione del pacchetto o della sua ultima revisione. La versione potete scriverla nella maniera che preferite. La breve descrizione deve essere una breve frase del tipo "Macro di Mario Rossi per comporre il manuale di tricotetratomia".

Dopo queste due righe potete inserire tutte le definizioni che volete e/o richiamare i pacchetti che volete; terminate il file con `\endinput`.

L'argomento delle definizioni verrà trattato nel capitolo 19.

6.7 Non modificare i pacchetti esistenti

Alcuni utenti di LaTeX credono di poter risolvere i loro problemi compositivi andando a modificare i file di classe o i pacchetti presenti nella propria distribuzione del sistema TeX o scaricati da CTAN.

Questo deve essere assolutamente evitato! Non lo si ripeterà mai abbastanza, ma questa pratica, oltre ad essere vietata dalla licenza a cui sono sottoposti quasi tutti i pacchetti esistenti, è una pratica autolesionista.

I modi corretti di procedere sono i seguenti.

1. Se le modifiche da apportare alla classe o al file di stile sono numerose, il modo migliore di procedere è quello di copiare il file in una cartella dell'albero personale `.../tex/latex/`⟨*cartella*⟩. Il nome di questa ⟨*cartella*⟩ verrà scelto in modo mnemonico, ma diverso dai nomi delle cartelle già esistenti; il file copiato verrà ribattezzato con un altro nome; per esempio se si vuole modificare la classe *book* si copierà il file `book.cls` nella propria ⟨*cartella*⟩ dandole, per esempio, il nome `tttomia.cls`.

   In questo nuovo file cambiate il contenuto di `\ProvidesClass`, per esempio modificate la dichiarazione

   ```
   \ProvidesClass{book}
                [2004/02/16 v1.4f Standard LaTeX document class]
   ```

   nel modo seguente

   ```
   \ProvidesClass{tttomia}
                [2007/03/10 v1.0
            Mario Rossi: classe per il libro di TricoTetraTomia]
   ```

   In questo nuovo file apportate tutte le modifiche che desiderate e aggiungeteci tutti i comandi che volete usare; questo siete liberi di farlo secondo la licenza delle classi, ma non siete liberi di modificare il file originale senza cambiargli il nome.

   Nello stesso tempo voi potete usare la classe standard in qualunque altra circostanza; questa inoltre vi potrà servire per seguire la stessa strada, con nomi diversi, se volete adattare la classe *book* alla composizione di un altro libro.

   Lo stesso procedimento può essere seguito per modificare i file di stile.

2. Potete modificare solo qualche comando che vi interessa in modo particolare; copiate dal file di classe originale o dal file di stile i comandi che volete modificare e incollateli nel vostro file `mymacros.sty` di comandi e macro personali; sostituite il comando `\newcommand` con `\renewcommand` e fate operazioni simili per le definizioni degli ambienti. Nel corpo della definizione inserite tutte le modifiche che volete.

Ricordate che i comandi definiti dentro i file di classe o di stile originali contengono spessissimo la 'lettera' @; è lecito usare questo segno come se fosse una lettera dell'alfabeto solo all'interno dei file di classe o di stile; quindi anche all'interno del vostro file `mymacros.sty`. State attenti però a non combinare pasticci; modificare o ridefinire comandi che contengano @ nel loro nome vi espone al rischio di modificare un comando di sistema in modo errato e tutto il vostro sistema TEX potrebbe diventare inutilizzabile; la riparazione è semplicissima,

perché vi basta tornare sui vostri passi eliminando dal vostro file le modifiche o le definizioni che potrebbero avere sconvolto il sistema, e poi potete ricominciare con più attenzione stando bene attenti a che cosa ridefinite o modificate. Usate sempre i comandi `\newcommand` e `\renewcommand` che controllano la precedente esistenza di comandi con lo stesso nome; potete usare per queste verifiche anche i comandi di sistema `\@ifdefinable` e `\@ifundefined`; le rispettive sintassi sono le seguenti:

```
\@ifdefinable{⟨comando⟩}{⟨azione⟩}
\@ifundefined{⟨nome del comando⟩}{⟨azione⟩}
```

Il comando `\@ifdefinable` verifica che il ⟨*comando*⟩ che si vorrebbe definire sia definibile; cioè che (*a*) non sia stato già definito, (*b*) sia diverso da `\relax`, (*c*) non cominci con `\end`. Se sono verificate queste condizioni, allora si può eseguire l'⟨*azione*⟩, verosimilmente la definizione vera e propria del ⟨*comando*⟩.

Invece `\@ifundefined` verifica che il ⟨*nome del comando*⟩ (il comando privato dell'iniziale backslash) sia davvero una entità priva di definizione, e se è verificata questa condizione esegue l'⟨*azione*⟩, verosimilmente la definizione.

Evitate di usare i comandi primitivi di definizione come `\def`, `\edef`, `\gdef` e `\xdef`, di cui si parlerà più avanti nel capitolo 19, perché questi comandi eseguono le definizioni in modo incondizionato e rischiate di azzoppare il sistema. Inoltre voi vedrete che nei file di classe e di stile questi comandi sono usati spessissimo; lasciateli dove sono ma non usateli voi stessi.

Per le definizioni di nuovi comandi o per la modifica di comandi esistenti bisogna consultare il capitolo 19. Tuttavia la raccomandazione più importante è quella di procedere lentamente con le 'acrobazie' consentite dal linguaggio di programmazione interpretabile da pdftex. Documentatevi adeguatamente e procedete per gradi; imparare il linguaggio di programmazione di pdftex è alla portata di tutti, ma richiede pazienza, attenzione, precisione e costanza.

Non si raccomanderà mai abbastanza: prima di mettersi a creare macro difficili, tortuose, inaffidabili, si cerchino prima le documentazioni dei pacchetti esistenti. A questo scopo è molto utile essersi caricati una distribuzione completa del sistema TEX. Dopo aver esaminato la documentazione si studino i pacchetti esistenti, anche se all'inizio sembrerà estremamente arduo capirci qualche cosa; ci si abitua con il tempo e con la pazienza. Poi se nonostante tutto si desidera produrre delle macro personali diverse da quelle che si trovano negli archivi, oppure veramente nuove, allora si disporrà di tutti i ferri del mestiere per cimentarsi in prima persona e probabilmente le macro prodotte non saranno affatto difficili, tortuose o inaffidabili.

Un amico che già conosca il linguaggio di sicuro è un grande aiuto, ma, come viene scritto in una delle citazioni riportate alla fine di ogni capitolo nel TEXbook[2]:

[2]Se non riesci a risolvere un problema, puoi sempre andare a leggere la soluzione [puoi sempre chiedere ad un amico]. Ma per favore, prova prima a risolverlo da solo; imparerai di più e più in fretta.

If you can't solve a problem,
you can always look up the answer.
But please, try first to solve it by yourself;
then you'll learn more and you'll learn faster.

— Donald E. Knuth, The TEXbook (1983)

La raccomandazione del padre del sistema TEX deve essere presa con grande serietà.

Capitolo 7

LaTeX: testi speciali

I testi speciali sono quelli che devono venire composti in modo diverso dal resto del testo; in particolare sono speciali quei testi da comporre interrompendo il testo principale o quelli da scrivere in parti della pagina diverse da quelle dove si troverebbero normalmente.

I testi che interrompono la composizione normale si chiamano *fuori testo* o, in inglese, in *display*. I brani di testo fuori sequenza sono generalmente costituiti dalle note a piè di pagina, da quelle a margine e da quelle alla fine del documento o di una suddivisione importante del documento, come, per esempio, alla fine dei capitoli.

Anche la matematica può essere composta in display, ma di questo si parlerà più avanti.

7.1 Che cosa sono i testi in display

I testi in display sono quelli 'in vetrina', quelli messi in evidenza in qualche modo o per il loro contenuto o per quello che rappresentano; si distinguono:

- le citazioni
- gli elenchi
- le descrizioni
- le liste bibliografiche

e verranno ora descritti una categoria alla volta.

7.2 Le citazioni

LaTeX offre tre ambienti per mettere in evidenza dei testi citati:

1. le citazioni brevi,
2. le citazioni lunghe,
3. le poesie.

7.2.1 Le citazioni brevi

Una citazione di un testo altrui che consista in poche parole può essere eseguita all'interno del testo racchiudendola fra virgolette. Le virgolette si indicano con i segni `‘‘` per le virgolette aperte e con `’’` per le virgolette chiuse. Siccome la tastiera italiana manca del tasto per segnare l'accento grave, bisogna appoggiarsi alle potenzialità dello *shell editor* per inserire questi segni mancanti, oppure bisogna conoscere i codici scrivibili nel file d'entrata mediante apposite combinazioni di tasti. Le virgolette caporali sono più facili da immettere quando si usa la codifica `T1` per il font di composizione del documento (vedi il capitolo 26); basta scrivere `<<` per i caporali aperti e `>>` per i caporali chiusi; si ottiene « ». Solo che i caporali sono usati meno frequentemente per le citazioni mentre sono usati più frequentemente per i dialoghi.

Con l'opzione *italian* di `babel`, indipendentemente dalla codifica usata per il testo di uscita, si può ricorrere alle funzionalità del carattere attivo `"` e scrivere `"<` per i caporali aperti e `">` per i caporali chiusi. Inoltre si può scrivere `""` per sfruttare le potenzialità del carattere attivo `"` che introduce in questo caso le doppie virgolette alte aperte “ come se si fossero potuti introdurre da tastiera due accenti gravi in sequenza.

A questo proposito merita ricordare che l'opzione *italian* per `babel` può fare diverse cose oltre a comporre in italiano; in particolare definisce le doppie virgolette `"` attive, definisce il comando `\unit` per inserire e unità di misura, e definisce la virgola in matematica come un carattere intelligente che scopre da solo (quasi sempre correttamente) se svolge el funzioni di separatore decimale o di segno di interpunzione. Ma, si badi bene, nessuna di queste funzionalità è disponibile se non le si attiva; per attivarle si usano i seguenti comandi:

```
% prima di \begin{document} ma dopo la chiamata di babel
\setISOcompliance % per attivare \unit
\setactivedoublequote % per attivare le doppie virgolette
% dopo \begin{document}
\IntelligentComma % per attivare al virgola intelligente
\NoIntelligentComma % per disattivare al virgola intelligente
```

Solo la “virgola intelligente” può essere attivata o disattivata; le altre due funzionalità sono disattive per impostazione predefinita, ma una volta attivate restano tali. Tutto ciò serve per evitare conflitti con altri pacchetti. In particolare ci sono diversi altri pacchetti che definiscono `\unit`; se uno di quei pacchetti viene caricato prima di `\begin{document}`, il comando `\unit` definito da `babel` con

l'opzione *italian* resta inerte mentre resta attivo quello del pacchetto invocato. Le doppie virgolette attive a quanto pare danno fastidio ad alcuni utenti che fanno uso del pacchetto *xypic*. La virgola intelligente può interferire saltuariamente con il funzionamento di alcune macro matematiche, ma di solito una volta attivata, non dà problemi di sorta; nel caso si verifichino dei problemi la si può disattivare. Non si ripeterà questo lungo inciso, ma si raccomanda di prendere nota che questo comportamento è disponibile dal 2013 e documenti composti in precedenza potrebbero avere bisogno di quei comandi di attivazione per poter essere composti come prima.

A questo proposito va segnalato che le virgolette aperte “ e quelle chiuse ” sono asimmetriche; in tipografia non bisogna mai usare le doppie virgolette simmetriche, anche se noti word processor lo fanno. Per questo motivo non bisogna mai usare il segno `"` per inserire le virgolette. Anzi con l'opzione per l'italiano di *babel* il segno `"` è attivo e quindi si comporta come una istruzione. Precisamente se esso è seguito da un qualunque carattere alfabetico esso inserisce un punto di possibile cesura e questa possibilità torna molto comoda per indicare dove andare a capo in quelle parole composte dove si vuole evitare la sillabazione fonetica prescritta dalle grammatiche e dalle norme tipografiche, ma si vuole usare la sillabazione etimologica: se si scrive `macro"istruzione` le due parole `macro` e `istruzione` sono trattate separatamente ai fini della sillabazione, ma una eventuale cesura viene esplicitata all'occorrenza anche fra la ‘o’ e la ‘i’ se la cesura in fin di riga lo richiede. Allo stesso modo `"/` consente di andare a capo dopo la barra quando si scrive, per esempio, `modulazione"/demodulazione` che viene composto come modulazione/demodulazione; in fin di riga modulazione/ demodulazione.

Tornando alle virgolette, siccome con la tastiera italiana l'inserzione delle virgolette aperte è problematica, se si usa *babel* con l'opzione per l'italiano, basta scrivere `""` per ottenere “.[1]

7.2.2 Le citazioni lunghe

Le citazioni più lunghe, anche consistenti in qualche riga di testo ma complessivamente senza superare la lunghezza di un capoverso, possono essere evidenziate mediante l'ambiente *quote* in questo modo:

```
\begin{quote}
⟨breve testo da citare⟩
\end{quote}
```

Il testo viene composto mettendolo in evidenza fra margini ristretti rispetto al testo normale.

[1] In generale sarebbe preferibile usare sempre una tastiera con il layout dei tasti corrispondente alla versione USA e poi usare le comode personalizzazioni per scrivere le lettere accentate, mancanti dalla tastiera USA. La stessa tecnica serve anche per le lettere maiuscole (“È arrivato” invece di “E' arrivato”) e per inserire gli accenti che non si usano in italiano, ma sono necessari per scrivere anche solo poche parole in altre lingue.

Per le citazioni più lunghe si usa l'ambiente *quotation*; la sintassi è:

```
\begin{quotation}
⟨testo lungo da citare⟩
\end{quotation}
```

La differenza rispetto all'ambiente *quote* è che in questo caso la citazione può contenere diversi capoversi; questi a loro volta, oltre ad essere composti fra margini ristretti, vengono composti con la prima riga rientrata del solito rientro di capoverso.

7.2.3 I versi

I versi sono una forma particolare di citazione, nel senso che viene composto un testo scritto da altri; tuttavia si tratta di un testo strutturato in versi e strofe; l'ambiente *verse* richiede che alla fine di ogni verso sia indicato l'ordine di andare a capo mediante il comando \\ e che ogni strofa sia trattata come un capoverso, cioè che venga lasciata una riga bianca fra una strofa e la successiva.

La sintassi è la seguente:

```
\begin{verse}
⟨verso⟩\\
⟨verso⟩\\
...
⟨verso⟩

⟨verso⟩\\
⟨verso⟩\\
...
⟨verso⟩
\end{verse}
```

Ecco, per esempio, il famoso sonetto di Dante Alighieri[2]:

> Tanto gentil e tanto onesta pare
> la donna mia quand'ella altrui saluta,
> ch'ogne lingua deven tremando muta,
> e li occhi no l'ardiscon di guardare.
>
> Ella si va, sentendosi laudare,
> benignamente d'umilta' vestuta;
> e par che sia una cosa venuta
> da cielo in terra a miracol mostrare.

[2]La mancanza di accenti, sostituiti con gli apostrofi di troncamento, sembra che sia una caratteristica dei manoscritti più accreditati. Probabilmente quegli apostrofi marcano la caduta di una o più lettere finali (apocope) rispetto alla versione latineggiante: umilta' al posto di umiltade; si' al posto di sic, puo' al posto di puote.

> Mostrasi si' piacente a chi la mira,
> che da' per li occhi una dolcezza al core,
> che 'ntender non la puo' chi no la prova;
>
> e par che de la sua labbia si mova
> uno spirito soave pien d'amore,
> che va dicendo a l'anima: Sospira.

viene composto con i comandi seguenti:

```
\begin{verse}
Tanto gentil e tanto onesta pare\\
la donna mia quand'ella altrui saluta,\\
ch'ogne lingua deven tremando muta,\\
e li occhi no l'ardiscon di guardare.

Ella si va, sentendosi laudare,\\
benignamente d'umilta' vestuta;\\
e par che sia una cosa venuta\\
da cielo in terra a miracol mostrare.

Mostrasi si' piacente a chi la mira,\\
che da' per li occhi una dolcezza al core,\\
che 'ntender non la puo' chi no la prova;

e par che de la sua labbia si mova\\
uno spirito soave pien d'amore,\\
che va dicendo a l'anima: Sospira.
\end{verse}
```

7.2.4 Brani in linguaggi speciali

Un tipo di citazione o di messa in evidenza di un testo è quello che serve per presentare brani di programmazione, come succede spesso in questo libro. Questo tipo di composizione richiede che nessun carattere sia speciale, che il font usato sia tale da rendere chiaro il fatto che il testo citato è un brano di programmazione, dove gli spazi e le rientranze possono avere un'importanza particolare, dove le linee troppo lunghe vanno a capo ma senza dividere in sillabe nessuna parola e dove anche gli 'a capo' devono essere rispettati senza preoccuparsi di giustificare il testo. Questo risultato si ottiene ricorrendo all'ambiente *verbatim*[3]. L'unica cosa che non si può riprodurre è proprio costituito dalla stringa di caratteri `\end{verbatim}` perché questa stringa, nella sua interezza serve per riconoscere la fine dell'ambiente. Questo testo abbonda di esempi di brani di programmazione; il più recente è quello usato per rappresentare il codice della poesia citata poco sopra. La sintassi dell'ambiente *verbatim* è la seguente:

[3]La parola latina 'verbatim' significa 'alla lettera'.

```
\begin{verbatim}
⟨testo da riprodurre verbatim⟩
\end{verbatim}
```

7.3 Gli elenchi

Gli elenchi sono generalmente di tre tipi: (*a*) in linea con il testo, come questo elenco; (*b*) in display con gli oggetti elencati distinti per mezzo di una numerazione, per cui questo tipo di elencazione si chiama *enumerazione*; e (*c*) in display dove però ogni elemento è evidenziato mediante un simbolo grafico, ma questo simbolo non è distintivo del particolare oggetto elencato; per cui questa è una semplice *elencazione*.

7.3.1 Le elencazioni in linea

Le elencazioni in linea sono quasi sempre composte come l'elencazione precedente, dove il segno distintivo di ogni oggetto elencato è una lettera in corsivo racchiusa fra parentesi tonde. L'elenco non è particolarmente evidenziato e la presenza delle lettere corsive è l'unico elemento che fa capire che si tratta di una elencazione; non è frequente, ma queste lettere corsive possono essere usate per richiamare l'oggetto identificato, nel qual caso devono essere riportate anche le parentesi, come quando ci si vuol riferire al caso (*a*).

7.3.2 Le enumerazioni

Gli elenchi numerati sono molto più strutturati anche in relazione alla possibilità di annidare un elenco dentro un altro. Le enumerazioni vengono realizzate ricorrendo all'ambiente *enumerate* con la seguente sintassi:

```
\begin{enumerate}
\item[⟨opzione⟩] ⟨oggetto testuale⟩
\item[⟨opzione⟩] ⟨oggetto testuale⟩
...
\end{enumerate}
```

Perciò quella che segue è una enumerazione, dove ogni voce costituisce almeno un periodo completo, cioè comincia con l'iniziale maiuscola contiene almeno un verbo e finisce con il punto fermo.

1. L'enumerazione viene composta dentro un ambiente *enumerate* come era facile prevedere.

2. Ogni oggetto da enumerare viene introdotto con il comando `\item`.

3. Ogni oggetto di una enumerazione può contenere un'altra enumerazione o una semplice elencazione.

4. Se per questo comando non si specifica nessuna opzione (e quindi non si scrivono nemmeno le parentesi quadre) l'oggetto viene numerato con un 'numero' progressivo.

5. Questo 'numero' può essere qualsiasi oggetto appartenente ad un sequenza ordinata di oggetti, tipicamente:

   - dei numeri arabi, o
   - dei numeri romani, oppure
   - delle lettere dell'alfabeto latino minuscolo, o
   - delle lettere dell'alfabeto latino maiuscolo.

   La classe del documento specifica quale tipo di numerazione viene eseguito per ogni livello di enumerazione annidata. Di default LaTeX con la sua classe `book` usa i numeri arabi, le lettere minuscole, i numeri romani, le lettere maiuscole.

6. Se invece viene espressa l'opzione, foss'anche una opzione 'nulla', allora invece del 'numero' viene scritta l'opzione.

⋆ Perciò si può fare riferimento simbolico agli elementi numerati per davvero, ma non si può fare riferimento, per esempio a questo elemento, perché non è numerato.

7. Come tutte le elencazioni, esse possono essere annidate una dentro l'altra, ma non si possono usare più di tre elencazioni annidate una dentro l'altra dentro una elencazione principale, non tanto perché la cosa sia tecnicamente impossibile, ma perché la struttura diventerebbe troppo complicata e il lettore ne rimarrebbe confuso.

7.3.3 Le elencazioni semplici

Le elencazioni semplici sono in display come le enumerazioni, ma ogni elemento elencato è solamente contrassegnato con un simbolo grafico uguale per tutti gli elementi dello stesso livello di elencazione. L'ambiente si chiama *itemize* e vale la sintassi seguente:

```
\begin{itemize}
\item[⟨opzione⟩] ⟨oggetto testuale⟩
\item[⟨opzione⟩] ⟨oggetto testuale⟩
...
\end{itemize}
```

Perciò:

- il primo livello di elencazione viene contrassegnato con dei pallini neri, ma

  - ⋆ se viene espressa una opzione, come in questo caso, il contrassegno è quanto viene specificato come opzione;

- Le elencazioni semplici possono essere annidate l'una dentro l'altra e il contrassegno cambia secondo il livello di annidamento. per esempio:

  - in questo secondo livello il contrassegno è costituito da un lineato medio;
  - si possono cambiare questi contrassegni agendo sulle definizioni che sono contenute nel file di classe;

  vedi anche l'elencazione annidata in una enumerazione nell'elemento 5 della pagina 137;

- in ogni caso si ricorre a queste elencazioni semplici quando gli oggetti da elencare non sono logicamente sequenziali;

- ci si ricordi anche che una elencazione semplice, anche se è in display, fa parte del capoverso che lo precede; quindi viene introdotta con una locuzione che termina con i due punti; ogni elemento, in quanto parte di un un unico capoverso, non termina col punto fermo, ma con una virgola o un punto e virgola oppure con una congiunzione; l'ultimo elemento elencato generalmente termina col punto fermo a meno che il capoverso non prosegua dopo l'elencazione.

7.3.4 Alcune osservazioni relative alle elencazioni

Nei paragrafi precedenti si sono descritte alcune elencazioni; la domanda ricorrente è la seguente: Quale tipo di punteggiatura viene usato fra un elemento e il successivo?

Per le elencazioni in linea si suppone che gli oggetti elencati siano brevi frasi, costituite al massimo da un solo periodo, quindi il punto e virgola sembra essere il segno di interpunzione più appropriato. Se non fosse così, allora sarebbe meglio costruire una elencazione in display.

Per le elencazioni in display, semplici o numerate, la punteggiatura può ancora essere un punto e virgola se gli oggetti sono brevi frasi, o frazioni di frase; ma se sono costituiti da periodi completi o da più periodi è necessario usare la punteggiatura che si userebbe anche senza evidenziare l'elencazione. Non necessariamente ogni oggetto elencato deve finire con la stessa punteggiatura, anzi, negli elenchi precedentemente usati a titolo di esempio si sono terminati alcuni oggetti senza punteggiatura ma con una congiunzione.

Nelle enumerazioni è possibile inserire il comando `\label` dentro gli oggetti a cui si vuole fare riferimento in modo simbolico, per esempio si può inserire nell'ultimo elemento dell'enumerazione usata come esempio:

```
\item \label{ele:annidato}Come tutte le elencazioni,
```

così che qui vi si possa fare riferimento mediante l'uso dei comandi `\ref` e `\pageref`, scrivendo

```
l'elemento~\ref{ele:annidato} nella pagina~\pageref{ele:annidato}.
```

ottenendo: "l'elemento 7 nella pagina 137."

7.4 Le descrizioni

Le descrizioni possono essere viste come elencazioni particolari nelle quali il segno distintivo non è né un numero né un simbolo, ma è una parola o una locuzione di cui si fornisce una descrizione o una spiegazione nel resto del testo dell'oggetto; la sintassi dell'ambiente *description* è la seguente:

```
\begin{description}
\item[⟨locuzione⟩] ⟨descrizione⟩
\item[⟨locuzione⟩] ⟨descrizione⟩
...
\end{description}
```

Benché la ⟨*locuzione*⟩ sembri facoltativa, essa in effetti è logicamente necessaria, perché non avrebbe nessun senso dare una ⟨*descrizione*⟩ di... nulla.

Questo ambiente è particolarmente indicato per dare una serie di definizioni, oppure per scrivere un glossario.

Esercizio 7.1 Continuando l'esercizio 5.1 si cominci un nuovo capitolo senza specificare l'asterisco, ma dandogli un titolo a piacere; si componga poi un ambiente *description* nel quale si danno le definizioni di avanzamento di riga, di corpo e di interlinea.

7.5 Le liste bibliografiche

Le liste bibliografiche sono dei particolari elenchi dove i riferimenti bibliografici sono etichettati con numeri o stringhe che possono essere richiamati simbolicamente nel corso del testo. Va subito detto che qui si parla dell'elenco bibliografico standard, cioè quello che si ottiene utilizzando l'ambiente *thebibliography* con la sintassi seguente per comporre l'elenco dei riferimenti bibliografici:

```
\begin{thebibliography}{⟨stringa⟩}
\bibitem[⟨richiamo⟩]{⟨chiave⟩} ⟨riferimento bibliografico⟩
\bibitem[⟨richiamo⟩]{⟨chiave⟩} ⟨riferimento bibliografico⟩
...
\end{thebibliography}
```

dove

⟨*stringa*⟩ è una stringa di caratteri, generalmente una stringa di cifre, ma il significato numerico è irrilevante; serve all'ambiente per prendere le misure per la lunghezza del campo da lasciare sul margine sinistro dell'elenco al fine di incolonnare correttamente tutte le voci della bibliografia.

⟨*richiamo*⟩ è una parola o una stringa di lettere e numeri che permette di identificare il particolare riferimento bibliografico all'interno del testo, oltre che nel campo identificativo dell'elenco; se non si specifica niente, (nemmeno le parentesi quadre), il riferimento di default è costituito da un numero, progressivo all'interno dell'elenco, racchiuso fra parentesi quadre. Questo è il modo di citazione più frequente negli articoli scientifici che si riferiscono a scienze nelle quali si fa un grande uso del linguaggio matematico.

Nelle discipline umanistiche è più frequente il modo di citazione 'autore-anno', per cui il ⟨*richiamo*⟩ è appunto costituito dal cognome del (primo) autore seguito dall'anno di pubblicazione dell'opera citata e, in caso di ambiguità, da una lettera progressiva. Ecco allora che la possibilità di stabilire con precisione il nome dell'autore e l'anno di pubblicazione dell'opera citata diventa un ottimo modo per specificare il ⟨*richiamo*⟩.

⟨*chiave*⟩ è il nome simbolico da usare sia nell'eventuale database bibliografico, sia come argomento obbligatorio del comando `\bibitem`, sia come argomento del comando `\cite`, che serve per citare il lavoro all'interno del documento.

L'ambiente *thebibliography*, nella classe `book`, provvede a iniziare una nuova pagina, a scrivere una intestazione di capitolo non numerato, ma intitolata Bibliography, oppure Bibliografia, oppure..., a seconda della lingua di default.

Se questo ambiente contenesse una voce del genere:

```
\bibitem{Lamp94} \textsc{Leslie Lamport},
\textit{A document preparation system --- \LaTeX\ ---
User's guide and reference manual}, Addison Wesley,
Reading, Mass., 2nd ed., 1994.
```

nella bibliografia apparirebbe una voce:

[12] Leslie Lamport, *A document preparation system — LaTeX — User's guide and reference manual*, Addison Wesley, Reading, Mass., 2nd ed., 1994.

e l'opera citata verrebbe citata con `\cite{Lamp94}`, per ottenere nel testo '[12]'.

Se invece nell'elenco bibliografico si fosse scritto:

```
\bibitem[\textsc{Lamport-94}]{Lamp94} \textsc{Leslie Lamport},
\textit{A document preparation system --- \LaTeX\ ---
User's guide and reference manual}, Addison Wesley,
Reading, Mass., 2nd ed., 1994.
```

nella bibliografia apparirebbe una voce

[Lamport-94] Leslie Lamport, *A document preparation system — LaTeX —*

User's guide and reference manual, Addison Wesley, Reading, Mass., 2nd ed., 1994.

e l'opera, citata sempre con la stessa chiave `\cite{Lamp94}`, apparirebbe nel testo come '[LAMPORT-94]'.

Facendo ricorso a pacchetti di estensione da richiamare con `\usepackage` nel preambolo, è possibile comporre questi elenchi in modo stilisticamente diverso; non solo, ma è possibile estrarre da un database bibliografico le informazioni necessarie mediante uno dei programmi BIBTEX o `biber`, così da avere un risultato dalla composizione omogenea oltre che semanticamente completo.

7.6 I riferimenti incrociati

Come si è visto nei paragrafi precedenti, è possibile riferirsi simbolicamente a qualunque oggetto numerato e alla pagina in cui compare, se questo tipo di informazione è sensato.

I comandi per ottenere questi riferimenti sono i seguenti e soddisfano alla sintassi qui di seguito indicata.

```
\label{⟨chiave⟩}
\ref{⟨chiave⟩}
\pageref{⟨chiave⟩}

\bibitem[⟨riferimento⟩]{⟨chiave⟩}
\cite[⟨informazioni aggiuntive⟩]{⟨chiavi⟩}
```

La ⟨*chiave*⟩ è una etichetta simbolica, possibilmente mnemonica (composta con qualunque sequenza di caratteri esclusa la virgola) con la quale si vuole identificare un oggetto. Come si è visto negli esempi precedenti, io compongo la chiave con una breve sigla che ricorda il tipo di oggetto, per esempio `fig:` per una figura, `ese:` per un esempio, `cap:` per un capitolo, `equ:` per una equazione, eccetera, seguito da un nome mnemonico che mi ricordi di che cosa si tratti.

Il comando `\label` con la sua ⟨*chiave*⟩, va inserito nel file sorgente *dopo* il comando che assegna un numero all'oggetto; i comandi `\ref` e `\pageref` con le loro chiavi vanno scritti nel testo nei punti dove si vuole fare riferimento all'oggetto; una precauzione utile dal punto di vista tipografico è quella di far precedere i comandi `\ref` e `\pageref` da una tilde, che in LATEX ha il significato di *spazio non separabile*, così che durante la composizione il motore TEX non spezzi la riga o non cambi pagina fra il nome che precede il riferimento e il riferimento stesso. Si scriverà, per esempio, `nell'equazione~\ref{equ:Pitagora}` così che TEX non spezzi la riga fra la parola 'equazione' e il numero che esso sostituirà alla ⟨*chiave*⟩ 'equ:Pitagora'.

Facendo uso del pacchetto di estensione *`varioref`* (da richiamare mediante il comando `\usepackage` nel preambolo) è possibile avere i riferimenti incrociati completi di numero e pagina in locuzioni del tipo 'la figura 5.13 nella pagina

seguente' oppure 'la figura 5.14 nella pagina 215' a seconda della distanza del richiamo dal punto in cui l'oggetto compare nel documento.

Per i riferimenti bibliografici il comando `\bibitem`, come si è visto, consente di definire una chiave e, facoltativamente, un richiamo. Il comando `\cite` accetta non solo una chiave, ma diverse chiavi separate da virgole; si potrebbe, per esempio, scrivere `\cite{lamp94,TexComp}` e ritrovarsi scritto nel testo '[12, 13]', cioè la citazione di entrambe le opere identificate mediante i loro numeri progressivi all'interno dell'elenco bibliografico.

Il comando `\cite` accetta anche un argomento facoltativo (che ha senso quando si cita una sola chiave) per indicare un posto preciso dell'opera citata; per esempio, se si scrive `\cite[cap.~4]{Lamp94}` si ottiene '[12, cap. 4]'.

Il vantaggio che questo metodo di citazione simbolico offre è che in caso di correzioni, per esempio cancellazioni o inserimenti di parte del testo, non è necessario correggere tutte le citazioni, ma ci pensa LaTeX a correggere tutti i valori che risultano modificati; bisogna solo lanciare LaTeX un paio di volte, finché nei messaggi finali non compare più il messaggio che 'alcuni riferimenti potrebbero essere stati modificati'.

7.7 Altri testi in display

Talvolta è necessario mettere in evidenza, separandolo dal resto del testo, un brano composto senza giustificazione, o meglio, solo allineato a sinistra, oppure solo allineato a destra, oppure centrato.

Per questo scopo sono disponibili i tre ambienti *flushleft*, *flushright* e *center*.

Si ricorda che il testo giustificato solo a sinistra è più facilmente leggibile di quello giustificato solo a destra e del testo in stile epigrafico con le righe centrate. Tuttavia per i titoli, per esempio, lo stile con le righe centrate è quasi sempre più indicato che non lo stile giustificato solo da un lato o giustificato da entrambi i lati.

Questi tre ambienti compongono il testo che essi racchiudono nella maniera specifica di ciascuno, ma tutti e tre hanno la caratteristica che il testo così composto risulta distanziato sopra e sotto mediante l'equivalente di una riga bianca (vuota).

In certe circostanze non è questo quello che si desidera, ma si vuole cambiare temporaneamente il tipo di giustificazione senza lasciare spazi bianchi prima o dopo il testo così composto.

Esistono i tre comandi (dichiarazioni) `\raggedright`, `\raggedleft` e, naturalmente, `\centering` che ottengono lo scopo desiderato; siccome non esiste un comando `\justify` che permetta di tornare al testo giustificato da entrambi i lati, è necessario limitare l'effetto di questi tre comandi mediante l'uso di un *gruppo*; un gruppo è una parte di testo intercalato a comandi racchiuso: (*a*) fra due parentesi graffe, oppure: (*b*) fra i comandi `\bgroup` ed `\egroup`, oppure: (*c*) fra i comandi `\begingroup` e `\endgroup`, oppure: (*d*) all'interno di ambienti delimitati da `\begin{...}` e `\end{...}`.

Tuttavia bisogna ricordare che l'interprete TEX esegue la composizione di ciascun capoverso, e quindi la sua giustificazione specifica, solo alla fine del capoverso stesso; per indicare esplicitamente la fine di un capoverso all'interno di un gruppo è meglio fare uso del comando `\par` (iniziale di *paragraph*, che in inglese indica il capoverso), che specifica proprio la fine di un capoverso. Perciò scrivere:

```
Andando lungo la strada egli vide una iscrizione
che annunciava:\\[1ex]
{\centering Qui visse Torquato Tasso durante il suo
soggiorno del 1605}\\[1ex] ma la cosa gli sembrava
strana, poiché ricordava che il Tasso era vissuto  nel~'500.
```

è sbagliato perché produce:

> Andando lungo la strada egli vide una iscrizione che annunciava:
>
> Qui visse Torquato Tasso durante il suo soggiorno del 1605
>
> ma la cosa gli sembrava strana, poiché ricordava che il Tasso era vissuto nel '500.

senza centrare il messaggio lapidario.
Invece scrivendo:

```
Andando lungo la strada egli vide una iscrizione
che annunciava:\par\vspace{1ex}
{\centering Qui visse Torquato Tasso durante il suo
soggiorno del 1605\par}\vspace{1ex}\noindent ma la cosa
gli sembrava strana, poiché ricordava che il
Tasso era vissuto  nel~'500.
```

si ottiene correttamente:

> Andando lungo la strada egli vide una iscrizione che annunciava:
>
> Qui visse Torquato Tasso durante il suo soggiorno del 1605
>
> ma la cosa gli sembrava strana, poiché ricordava che il Tasso era vissuto nel '500.

Nell'esempio precedente, oltre all'illustrazione dell'uso di `\centering` e del comando `\par`, vengono usati diversi altri comandi non ancora descritti, ma che vale la pena di commentare qui.

`\\` serve per andare a capo sia nel mezzo di un testo, sia quando si compongono matrici matematiche o tabelle; questo comando accetta un argomento facoltativo, racchiuso fra parentesi quadre, che indica quanto spazio verticale lasciare dopo essere andati a capo. `\\[1ex]` vuol dire: 'vai a capo e

lascia uno spazio verticale pari a 1 ex, cioè pari all'altezza di una lettera 'x' nel font corrente'. Esiste anche la variante `\\*`, che continua ad accettare l'argomento facoltativo, dove l'asterisco impone a TeX di non andare a pagina nuova con la nuova riga.

Vale la pena di sottolineare che `\\` serve per andare a capo alterando la giustificazione di quella riga all'interno di un capoverso; è errato lasciare una riga vuota e cominciare una altro capoverso con `\noindent`, che solo ad un occhio distratto sembra produrre lo stesso effetto; è altrettanto errato, al contrario, cominciare un altro capoverso scrivendo `\\\indent`, perché si produce la rientranza (*indent* in inglese) ma non si comincia affatto un nuovo capoverso. Con certi stili grafici e/o certi ambienti la rientranza potrebbe essere nulla; con certi altri stili grafici e/o ambienti la separazione fra i capoversi non è indicata dalla rientranza, ma da una visibile interlineatura.

`\vspace` serve per inserire uno spazio verticale dopo la fine di un capoverso o, detto in termini più tecnici, quando TeX è in modo verticale. L'argomento indica quanto spazio lasciare; siccome questo spazio viene perso all'inizio di una pagina, la forma asteriscata `\vspace*` consente di mantenere lo spazio specificato anche all'inizio della pagina.

`\noindent` serve per evitare il rientro all'inizio di un capoverso.

Attenzione! Sarebbe desiderabile non fare mai uso dei comandi appena descritti; questi non sono parte del mark-up, ma del disegno grafico; sono comandi prescrittivi, non comandi descrittivi. L'autore/compositore dovrebbe attenersi al mark-up descrittivo la maggior parte del tempo. È costretto a ricorrere a questi 'sotterfugi' quando la situazione non prevede qualche ambiente che gli consenta di ottenere la composizione nella maniera desiderata. Piuttosto egli farebbe meglio a definire un nuovo ambiente con le caratteristiche desiderate, ma questo è un argomento che verrà visto molto più avanti.

Nota bene! Al contrario è opportuno usare `\centering` all'interno degli ambienti *figure* e *table* per centrare l'inclusione dell'illustrazione o della tabella da flottare; non occorre servirsi di `\par` perché quegli ambienti contengono già tutte le informazioni per poter gestire questa situazione. Se si usasse invece l'ambiente *center* per centrare l'illustrazione o la tabella, questo ambiente inserirebbe dell'ulteriore spazio bianco prima e dopo la sua apertura e la sua chiusura, per cui l'oggetto flottante risulterebbe poi preceduto e seguito da troppo spazio bianco e/o la didascalia risulterebbe troppo spaziata.

Infatti l'uso di `\centering` per specificare la collocazione orizzontale delle figure e delle tabelle all'interno dei loro rispettivi ambienti flottanti consente di evitare l'inserimento di spazi bianchi in eccesso; per l'uso dettagliato di questi ambienti flottanti si rimanda al paragrafo 9.2; qui li si simulerà per

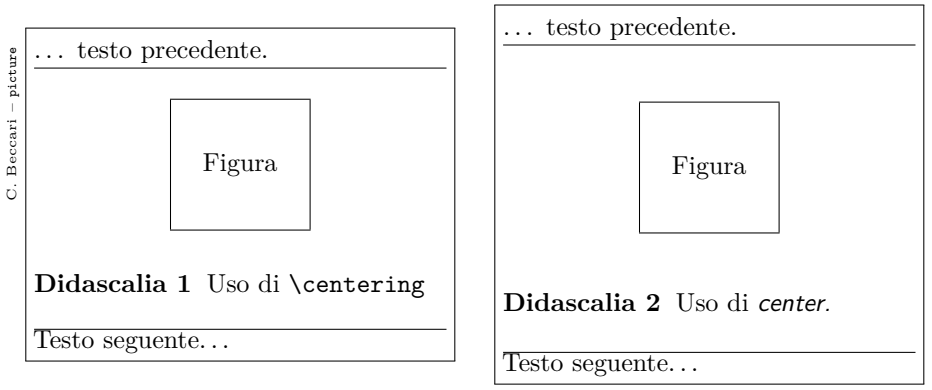


Figura 7.1: Effetto dell'uso del comando `\centering` rispetto all'ambiente *center*.

mostrare la differenza fra il risultato ottenibile con il comando `\centering` rispetto a quello ottenibile con l'ambiente *center*; in entrambi i casi vengono collocati due filetti orizzontali per marcare la differente spaziatura verticale.

Come si vede nella figura 7.1, l'oggetto (rappresentato in questo esempio da un semplice rettangolo) viene inserito fra il testo precedente e il testo seguente, insieme alla sua didascalia, distanziato dagli spazi previsti dal file di classe quando si usa `\centering`. Quando invece si usa l'ambiente *center*, questo inserisce ulteriori spazi e l'oggetto e la sua didascalia risultano contornati da troppo spazio bianco.

7.8 Le note

LaTeX consente di scrivere due tipi di note, quelle a piè di pagina e quelle composte nel margine della pagina. Per scrivere note raccolte assieme alla fine di un capitolo, di un articolo, di un libro, bisogna ricorrere al pacchetto esterno *endnotes*[4].

7.8.1 Le note in calce

Per scrivere una nota in calce basta usare il comando `\footnote` con la seguente sintassi:

`\footnote[`⟨*richiamo*⟩`]{`⟨*testo*⟩`}`

Il ⟨*richiamo*⟩ di default è un numero collocato in posizione di apice; talvolta può essere un simbolo come un asterisco, una spada, una doppia spada, eccetera; la numerazione delle note normalmente ricomincia da 1 con l'inizio di un nuovo

[4]Non si confonda il pacchetto *endnotes*, formato dal file `endnotes.sty`, con il file `endnote.tex`, che è un file di estensione per Plain TeX, non per LaTeX.

capitolo, ma la cosa dipende dalla classe del documento. Il ⟨*testo*⟩ della nota non richiede commenti, salvo che si tratta di un 'argomento mobile', e quindi certi comandi 'fragili' non possono essere usati all'interno del ⟨*testo*⟩. Il lettore non si preoccupi dei comandi fragili e degli argomenti mobili; sono cose abbastanza rare quando si scrive un documento 'normale'; in questo testo, che descrive i comandi di LaTeX alcune cose non sono 'normali', proprio perché si tratta di comandi; perciò quando alcuni comandi comparivano nelle note si è dovuto ricorrere a trucchetti vari per poterli scrivere alla lettera, senza farli eseguire.

Le note sono testi speciali, perché vengono composte e trattenute in memoria fino a quando non viene composta la pagina nella quale esse devono comparire; LaTeX fa il possibile affinché le note non siano spezzate fra pagine successive e che ognuna sia collocata al piede della pagina nella quale compare il richiamo. È abbastanza raro che LaTeX non riesca a sistemare la nota in calce senza spezzarla, ma può succedere. Non è colpa di LaTeX, ma del testo nel quale la nota è richiamata e del testo della nota. Talvolta, cambiando l'uno o l'altro il problema si risolve da solo.

È possibile attribuire una etichetta ad una nota mediante il comando `\label`; e a questa etichetta si può fare riferimento per citare la stessa nota in punti diversi del testo.

All'occorrenza si può fare uso di un comando separato per inserire il richiamo e per scrivere il testo; i comandi seguono la sintassi seguente:

```
\footnotemark[⟨richiamo⟩]
\footnotetext[⟨richiamo⟩]{⟨testo⟩}
```

nei quali il ⟨*richiamo*⟩ è facoltativo; se si fa uso dell'argomento facoltativo sia in questi comandi sia in `\footnote` non viene incrementato il contatore delle note e quindi l'attribuzione di una etichetta mediante `\label` non funziona.

Questi due comandi sono prescrittivi, non descrittivi, perché scendono nel dettaglio di come fare apparire le note nel testo composto. Questi comandi, anche se in generale è bene che siano evitati, tuttavia possono essere necessari quando il procedimento normale di composizione di LaTeX non è adeguato al compito. Per esempio non è possibile mettere note dentro alle tabelle; esistono pacchetti di estensione per superare questi limiti, ma se non capita sovente di inserire note nelle tabelle, può valere la pena di usare `\footnotemark` specificando il richiamo dentro una casella della tabella e di usare `\footnotetext` con lo stesso richiamo dopo il comando `\caption` per comporre la didascalia.

Pertanto le note da attribuire alle tabelle vanno composte come mostrato nel seguente modello. Ci si ricordi di specificare una ⟨*larghezza*⟩ adatta a contenere la tabella. Nello stesso tempo non si perda di vista la possibilità di specificare gli argomenti facoltativi sia di `\footnotemark` sia di `\footnotetext`, perché essi devono venire composti con lo stile che LaTeX usa all'interno delle *minipage* (richiami mediante apici letterali in corsivo), ma che, essendo l'ambiente *tabular* composto dentro una scatola, potrebbero non venire stampati correttamente. Esistono ovviamente pacchetti che consentono di eseguire quanto esemplificato

qui in modo 'automatico', ma anche Lamport consiglia nel suo manuale di ricorrere all'uso di `\footnotemark` e `\footnotetext` proprio in questi casi.

```
\begin{table}
\begin{minipage}{⟨larghezza⟩}
\centering
\begin{tabular}{...}
... & ... & ... \\
...
... & ... & ...\footnotemark \\
...
\end{tabular}
\caption{⟨didascalia⟩}\label{⟨chiave⟩}
\footnotetext{⟨testo della nota⟩}
\end{minipage}
\end{table}
```

7.8.2 Le note marginali

Se i margini del documento sono abbastanza ampi, questi possono accogliere le note a margine, talvolta chiamate in latino *marginalia*. Il comando per produrle è `\marginpar` e segue la sintassi:

```
\marginpar[⟨nota di sinistra⟩]{⟨nota di destra⟩}
```

Anche le note marginali sono testi che vengono trattenuti in memoria finché la pagina in cui compaiono non viene completamente composta; a seconda del tipo di documento, della classe, dello stile grafico, dell'ampiezza dei margini, la nota marginale può venire collocata nel margine esterno o in quello interno; se però viene collocata a destra del testo, viene composto il testo della ⟨*nota di destra*⟩, altrimenti il testo della ⟨*nota di sinistra*⟩, se è stato specificato, altrimenti il testo della ⟨*nota di destra*⟩ viene usato anche per la ⟨*nota di sinistra*⟩.

Si osservi che se si compone a due colonne, le note relative alla colonna di sinistra vengono collocate a sinistra della colonna e, corrispondentemente le note relative alla colonna di destra vengono collocate a destra della colonna. Componendo ad una colonna le note vengono collocate sempre nel margine esterno, quindi a destra del testo nelle pagine di destra e a sinistra del testo nelle pagine di sinistra. Sembra complicato, ma a pensarci bene è del tutto logico. È vero che con l'apposito comando `\reversemarginpar` questo comportamento può venire scambiato, ma in generale si suppone che questa scelta sia fatta nel file di classe e il compositore non debba preoccuparsene.

Le note marginali vengono composte in colonne strette, con una giustezza che raramente supera i 40 mm; con una giustezza così piccola è possibile che la composizione a pacchetto (giustificata da entrambi i lati) produca una composizione con ampi spazi bianchi fra le parole e con parole difficili da dividere in sillabe che sporgono fuori dalla giustezza. In questi casi sarebbe meglio specificare per

la nota di destra una composizione giustificata solo a sinistra; basta usare il comando `\raggedright`. Per le note di sinistra, per una questione di simmetria, sarebbe quindi desiderabile specificare un testo giustificato solo a destra mediante il comando `\raggedleft`. La cosa è evidentemente possibile se si fa uso dell'argomento facoltativo descritto nella sintassi di `\marginpar`. Tuttavia la lettura di un testo giustificato solo a destra può risultare meno agevole di un testo giustificato solo a sinistra; ecco perché, pur essendo possibile, raramente si specifica una composizione giustificata solo a destra, anche se i testi delle note marginali sono solitamente molto brevi e quindi non sono faticosi da leggere anche se fossero giustificati solo a destra.

Questa è una nota marginale che si sviluppa su più righe.

Questa è una seconda nota marginale troppo vicina alla nota precedente.

Le note marginali sono collocate nel margine che compete loro mantenendo la loro prima riga allineata con la riga del testo alla quale esse si riferiscono; se due note marginali compaiono troppo ravvicinate e ci fosse il rischio che si sovrappongano, allora LaTeX provvede a spostare in basso le note che potrebbero interferire con le note precedenti assicurando uno spazio di separazione specificato nel file di classe; come si vede nelle due note qui a margine, la seconda, specificata nella riga dove si dice "compaiono troppo ravvicinate...", risulta troppo ravvicinata alla prima ed è stata leggermente abbassata.

Le note marginali vengono usate spesso negli scritti letterari, specialmente se si tratta di note brevi e frequenti; è raro incontrare note marginali in scritti tecnici.

7.9 Un esempio specifico di testi speciali

Scrivendo libri didattici capita di inserire degli esercizi; direi anzi che è buona norma inserire molti esercizi, alcuni svolti, altri da svolgere; alcuni con le soluzioni che li seguono direttamente, altri che hanno le soluzioni in un'apposita appendice, altri ancora senza soluzione. Nei testi statunitensi, spesso vengono fornite in appendice solo le soluzioni degli esercizi numerati con un numero dispari, lasciando gli esercizi pari agli studenti, affinché imparino a lavorare in modo autonomo e imparino anche ad analizzare la soluzione per verificarne la correttezza.

Gli esercizi spesso dopo aver presentato una panoramica del problema da risolvere, propongono diverse domande, spesso strutturate gerarchicamente; le soluzioni, se fornite, devono rispettare lo stesso schema. Riconosciamo subito che queste domande gerarchiche formano tipicamente una lista strutturata, quindi per questi insiemi di domande e risposte gli ambienti delle liste descritte nei paragrafi precedenti devono essere usati esplicitamente o implicitamente.

Ma c'è un altro problema: per chi compone il libro è importante che il testo dei ciascun esercizio sia immediatamente seguito dalla sua soluzione, cosicché se l'intero esercizio è da sostituire con un altro diventa quasi immediato selezionare l'intero blocco delle righe del testo da sostituire e cancellarlo, sostituendolo con un altro testo dell'esercizio e della sua soluzione. Per la validità didattica, invece, non è mai opportuno che la soluzione segua immediatamente l'enunciato dell'esercizio; infatti solitamente le soluzioni sono raccolte in una apposita appendice.

Certo non sarebbe troppo difficile comporre il testo sorgente delle soluzioni in un file separato e includere questo file verso la fine del testo. Ma ogni volta che si deve eseguire qualche correzione o sostituzione è facilissimo commettere errori e sostituire uno dei testi (enunciato o soluzione) con testi spaiati. Mi spiego: l'esercizio 3.15 è abbinato alla sua soluzione 3.15; ma se i due testi nei file sorgente sono distanti l'uno dall'altro potrebbe succedere di sostituire l'enunciato 3.15 con un altro enunciato, la cui soluzione, però viene sostituita per errore alla soluzione dell'enunciato 3.16.

È chiaro che le soluzioni e gli enunciati fanno riferimento le une agli altri mediante il meccanismo dei riferimenti incrociati resi con `\ref` e `\label` e le stesse chiavi. Ma è facile sbagliarsi, quindi è meglio ridurre le possibilità di errore mantenendo l'enunciato adiacente alla sua soluzione, magari nello stesso file da sostituire eventualmente con un altro file contenente un altro enunciato con la sua soluzione.

In questo modo le possibilità d'errore sono davvero minime.

Ma nasce il problema: come si fa a dilazionare la stampa della soluzione al punto giusto fra le appendici? La cosa non è facile per niente, non parliamo di eseguirla con un normale word processor. Ma con LaTeX si può fare.

Basta caricare il pacchetto *exercise*; questo pacchetto essenzialmente definisce due ambienti: *Exercise* e *Answer*, il primo per contenere l'enunciato dell'esercizio e il secondo per contenere la sua soluzione. Entrambi richiedono di inserire delle opzioni del tipo ⟨*chiave*⟩=⟨*valore*⟩. Perché i due ambienti si riferiscano l'uno all'altro, fra le opzioni di *Exercise* bisogna inserire `label={`⟨*etichetta*⟩`}` e fra le opzioni di *Answer* bisogna inserire `ref={`⟨*etichetta*⟩`}`, usando la stessa ⟨*etichetta*⟩. Naturalmente si possono inserire altre parole chiave che è meglio esaminare studiando attentamente la documentazione del pacchetto *exercise*.

Non si scende in ulteriori dettagli. Il concetto che si vuole evidenziare è che quando si compone il testo, il contenuto dell'ambiente *Exercise* viene composto subito nel file di uscita, mentre il contenuto dell'ambiente *Answer* viene composto dentro un registro della memoria del programma di composizione e ogni successiva soluzione di altri esercizi viene accodata a questo registro. Al momento di stampare le soluzioni, questo registro viene svuotato nel flusso di testo composto destinato al file di uscita. Semplice a dire, ma creare le macro del pacchetto *exercise* ha richiesto non poca immaginazione e competenza al suo autore.

Il seguente banale esercizio illustra quanto si può fare con questo pacchetto; ovviamente si possono specificare molte opzioni in modo da poter personalizzare l'aspetto dell'enunciato e, rispettivamente, l'aspetto della soluzione. si possono aggiungere anche alte informazioni, alcune delle quali rappresentano solo delle annotazioni per il compositore. L'intestazione dell'enunciato e quella della soluzione possono venire personalizzate in molti modi. In ogni caso essi possono essere localizzati in una lingua particolare se (*a*) il pacchetto *exercise* viene caricato dopo *babel*, e (*b*) se le definizioni nella lingua principale del documento sono già esistenti nel pacchetto. Nel momento di scrivere questo testo (maggio 2015) il pacchetto contiene le localizzazioni per l'inglese, il francese, lo spagnolo, l'italiano, l'olandese, il tedesco, il portoghese e il russo; l'autore attende dagli

utenti le traduzioni per altre lingue, come succede normalmente per il software libero.

Esercizio 1

Calcolare la lunghezza del secondo cateto in un triangolo rettangolo con il primo cateto che vale 12 cm e l'ipotenusa che vale 13 cm.

Il testo sorgente di questo esercizio e della soluzione è il seguente:

```
\begin{Exercise}[label={pitagora},difficulty={0}]
Calcolare la lunghezza del secondo cateto in un triangolo
rettangolo con il primo cateto che vale 12\unit{cm} e
l'ipotenusa che vale 13\unit{cm}.
\end{Exercise}

\begin{Answer}[ref={pitagora}]
Dal teorema di Pitagora sappiamo che:
\[a^2 + b^2 = c^2\]
dove $a$ e $b$ sono i cateti e $c$ è l'ipotenusa si ha:
\begin{align*}
12^2 + b^2 &= 13^2\\
144 + b^2  &= 169\\
\intertext{da cui:}
b^2        &= 169 - 144 = 25\\
b          &= \pm5\\
\intertext{e, avendo eliminato la soluzione negativa, la soluzione
           dell'esercizio è:}
b          &= 5\unit{cm}
\end{align*}
\end{Answer}
```

In questo esempio non si stampa subito la soluzione ma la si compone solo alla fine di questo capitolo mediante il comando `\shipoutAnswer`.

Soluzione dell'esercizio 1

Dal teorema di Pitagora:

$$a^2 + b^2 = c^2$$

dove a e b sono i cateti e c è l'ipotenusa si ha:

$$12^2 + b^2 = 13^2$$
$$144 + b^2 = 169$$

da cui:

$$b^2 = 169 - 144 = 25$$
$$b = \pm 5$$

e, avendo eliminato la soluzione negativa, la soluzione dell'esercizio è:

$$b = 5\,\mathrm{cm}$$

Capitolo 8

LaTeX: tabelle

8.1 Introduzione

Le tabelle sono oggetti strutturati, contenenti testo o formule o espressioni matematiche, formati da un certo numero di celle ordinate in senso orizzontale e in senso verticale; queste celle possono essere riquadrate mediante filetti; spesso ciascuna colonna di celle ha una prima cella il cui contenuto descrive quello delle celle sottostanti.

Comporre questo insieme di celle è sempre stato un grosso problema anche ai tempi in cui la composizione veniva eseguita a mano; ora i vari word processor e text processor consentono di lavorare con più facilità, ma senz'altro la composizione delle tabelle, specialmente delle tabelle ben composte, richiede molta pazienza e molta preparazione iniziale; se il compositore si fa uno schizzo, anche grossolano, della tabella, sa già in anticipo dove sorgeranno i maggiori problemi e quindi sa predisporre gli accorgimenti adeguati.

Qui si parlerà di tabelle 'testuali'; quelle con contenuto prevalentemente matematico si chiamano in generale *matrici* e verranno viste nel capitolo 14.

Distinguiamo subito due casi, cioè quello delle tabelle che non occupano più di una pagina, e le tabelle lunghe, che occupano più pagine. Le tabelle 'brevi', nonostante la loro brevità, sono oggetti ingombranti, perciò è meglio che il programma di elaborazione del testo, LaTeX nel nostro caso, sposti la tabella nel documento composto dove c'è sufficiente posto per collocarla, talvolta nella stessa pagina o persino nel punto stesso dove è stata definita, più spesso all'inizio o alla fine della pagina successiva, talvolta alla fine di un capitolo o dell'intero documento.

Per consentire questo trattamento LaTeX deve conservare in memoria la tabella composta, per poi scaricarla nel file di uscita alla prima occasione buona.

Questo modo di procedere implica due cose:

- le tabelle devono essere numerate per farvi riferimento mediante i loro numero;

- le tabelle troppo grandi possono bloccare la memoria perché vi vengono trattenute in quanto LaTeX non trova mai l'occasione buona per scaricarle nel file di uscita.

Il fatto che le tabelle siano numerate e che debbano essere fatte 'flottare' fino al punto più idoneo richiede un ambiente specifico di 'flottaggio', e implicano un comando per eseguire la didascalia, anche una semplice didascalia composta dal solo titolino.

8.2 Come far flottare una tabella

L'ambiente di flottaggio si chiama *table* e richiede la seguente sintassi:

```
\begin{table}[⟨codici di posizionamento⟩]
⟨didascalia e tabella vera e propria⟩
\end{table}
```

I ⟨*codici di posizionamento*⟩ sono i codici con i quali viene descritto il modo di collocare la tabella nel testo; di default questi codici sono `t`, `b` e `p`; i codici sono complessivamente i seguenti:

`h`	poni la tabella qui (*here* in inglese)
`t`	poni la tabella in testa alla pagina (*top*)
`b`	poni la tabella al fondo della pagina (*bottom*)
`p`	poni la tabella in una pagina di soli oggetti flottanti (*page*)
`!`	poni la tabella dove prescritto dagli altri codici di posizione senza essere troppo rigoroso con i parametri di posizionamento

Per maggiori dettagli si veda il paragrafo 9.2 dove sono illustrati vari metodi per controllare la posizione degli oggetti flottanti e dove si descrivono alcuni pacchetti che consentono di modificare gli ambienti standard e di definire nuovi ambienti flottanti.

8.3 Le didascalie

Quanto segue vale per tutte le didascalie, anche per quelle delle figure. Come impostazione predefinita nelle classi standard LaTeX pone la didascalia sotto alle tabelle; in Europa, e in Italia in particolare, viene raccomandato di inserire la didascalia prima della tabella a cui si riferisce. Per risolvere questo problema o si usano classi non standard, come *memoir*, oppure si usa il pacchetto *caption* impostandolo adeguatamente. Invece per le figure si preferisce la didascalia dopo l'illustrazione. Ovviamente questa è una affermazione generale, visto che le didascalie possono trovare diverse posizioni a seconda del tipo di illustrazioni o di tabelle e del tipo di didascalie. Nuovamente, per usare didascalie laterali

o composte con stili diversi da quello preimpostato con le classi standard, o si usano classi non standard o si usa il pacchetto di estensione *caption*.

La didascalia viene composta mediante il comando `\caption` secondo la seguente sintassi:

```
\caption[⟨didascalia breve⟩]{⟨didascalia⟩}\label{⟨etichetta⟩}
```

La ⟨*didascalia*⟩ è il testo a cui è stato premesso il titolino 'Tabella', o 'Table', o 'Tableau', o ..., a seconda della lingua in uso, seguito dal numero progressivo della tabella; questa parte viene inserita di default; il resto della didascalia può anche essere omesso, ma non sarebbe una buona idea. Il resto della didascalia può essere composto con un titolo seguito da un capoverso descrittivo. Questo capoverso può mancare; se c'è, viene terminato con il punto fermo. Se non c'è, il titolo viene terminato senza punto fermo, come tutti i titoli.

Non è il caso che l'intera ⟨*didascalia*⟩ vada a finire nell'elenco delle tabelle, al massimo ci andrà il titolo, ma non certo il capoverso descrittivo. Ecco, quindi, che la ⟨*didascalia breve*⟩ torna utile per essere inserita nell'elenco delle tabelle, eventualmente abbreviando ulteriormente il titolo inserito nella didascalia completa.

L'⟨*etichetta*⟩ introdotta con il comando `\label` serve per poter fare riferimento al suo numero (assegnato automaticamente da LaTeX) mediante un nome simbolico; è opportuno, quindi, che l'⟨*etichetta*⟩ sia scelta con qualche criterio mnemonico in modo che essa richiami il contenuto della tabella, piuttosto che un'altra caratteristica qualsiasi. Nel corso del testo la tabella viene richiamata per numero mediante l'uso del comando `\ref` e/o per numero di pagina mediante l'uso del comando `\pageref`, secondo la sintassi seguente:

```
\ref{⟨etichetta⟩}
\pageref{⟨etichetta⟩}
```

È opportuno fare uso del segno di legatura `~` per non staccare il riferimento della tabella dal nome che l'accompagna; si scriverà pertanto:

```
... la tabella~\ref{tab:popolazione} mostra che ...
```

per riferirsi alla tabella etichettata con `tab:popolazione` con la certezza che non potrà avvenire un fine riga o un fine pagina fra la parola 'tabella' e il suo riferimento numerico.

Per modificare lo stile compositivo delle didascalie esiste l'ottimo pacchetto di estensione *caption* mediante il quale si possono modificare tutti i possibili parametri che presiedono alla loro composizione.

8.4 Come comporre la tabella vera e propria

La tabella vera e propria viene composta mediante l'ambiente *tabular* che richiede la seguente sintassi:

Descrittore	Spiegazione
`l`	Cella col contenuto allineato a sinistra
`c`	Cella con il contenuto centrato
`r`	Cella con il contenuto allineato a destra
`p{`⟨*larghezza*⟩`}`	Cella contenente un blocco di testo giustificato e largo ⟨*larghezza*⟩
`|`	Filetto verticale
`@{`⟨*espressione*⟩`}`	@-espressione
`\vline`	Filetto verticale da sostituire a `|` dentro una @-espressione
`*{`⟨*numero*⟩`}{`⟨*descrittori*⟩`}`	Ripetizione di ⟨*numero*⟩ volte la stessa sequenza di ⟨*descrittori*⟩

Tabella 8.1: Descrittori delle colonne per le tabelle

```
\begin{tabular}[⟨allineamento⟩]{⟨descrittori delle colonne⟩}
⟨prima cella⟩&⟨seconda cella⟩&...\\
⟨prima cella⟩&⟨seconda cella⟩&...\\
...\\
⟨prima cella⟩&⟨seconda cella⟩&...\\
\end{tabular}
```

C'è anche la versione asteriscata *tabular** che obbedisce alla sintassi seguente:

```
\begin{tabular*}{⟨larghezza⟩}[⟨allineamento⟩]{⟨descrittori delle colonne⟩}
⟨prima cella⟩&⟨seconda cella⟩&...\\
⟨prima cella⟩&⟨seconda cella⟩&...\\
...\\
⟨prima cella⟩&⟨seconda cella⟩&...\\
\end{tabular*}
```

Per l'ambiente *tabular** si vedano i dettagli nel paragrafo 8.5; qui, per ora, concentreremo l'attenzione sull'ambiente *tabular*, ma ricordiamo che quasi tutto quello che verrà detto qui vale anche per le tabelle dell'ambiente *tabular**.

8.4.1 I descrittori delle colonne

I descrittori delle colonne sono esposti nella tabella 8.1.

Mentre tutto il resto sembra ovvio, è necessario spiegare che cosa sia una *@-espressione*; questa è una sequenza di comandi, di descrittori, di elementi testuali da mettere fra due celle adiacenti al posto del normale spazio fra le colonne. Anche il descrittore `|` potrebbe essere visto come un separatore fra colonne adiacenti (e lo è) ma non è una @-espressione; questa sostituisce *anche* lo spazio di separazione fra le celle adiacenti, e quindi fra le colonne. Per esempio,

si può notare che la tabella 8.1 ha i filetti orizzontali che sporgono un poco fuori dei limiti della cella di sinistra e di destra. Generalmente l'effetto estetico è gradevole, ma potrebbe essere necessario eliminare tale spazio; serve allora inserire come primo e ultimo descrittore di colonna una @-espressione 'vuota': `@{}`.

Il parametro facoltativo di ⟨*allineamento*⟩ serve per allineare sulla stessa riga almeno due tabelle; i rispettivi codici sono `t` (*top*), `b` (*bottom*) e `c` (*center* – preimpostato); si vedano i dettagli nella pagina 682.

Esercizio 8.1 Una @-espressione potrebbe essere usata per ripetere come separatore di colonna uno stesso segno; in una tabella che riportasse una colonna con valori numerici fratti e si volesse incolonnare questi numeri in modo che il separatore decimale sia incolonnato fra tutte le celle, indipendentemente dal numero delle cifre prima e dopo la virgola, si potrebbe usare la virgola come testo all'interno della @-espressione spezzando la colonna in due colonne adiacenti e incolonnare la parte intera in una colonna allineata a destra e la parte decimale in una colonna allineata a sinistra.

Predisporre una tabella con una colonna di testi descrittivi per ogni riga e una 'colonna' di numeri decimali incolonnata sulle rispettive virgole decimali.

L'esercizio precedente è utile per capire il meccanismo; per fortuna il pacchetto di estensione *`array`* consente già mediante un nuovo descrittore di incolonnare i numeri fratti sulla base del loro separatore decimale. Le estensioni che questo pacchetto consente sono tali e tante che converrebbe richiamarlo costantemente per ogni documento da comporre. Si rinvia alla documentazione di quel pacchetto per maggiori dettagli.

Il descrittore `p`, come già detto, permette di comporre il contenuto di una cella come un piccolo capoverso giustificato a pacchetto (da entrambi i lati); dato che le celle hanno una larghezza limitata e certamente minore di quella dell'intero testo, talvolta è consigliabile specificare `\raggedright` oppure, meno frequentemente, `\raggedleft`. Ci si ricordi che il comando `\\` serve sia per andare a capo in un capoverso, sia per terminare la riga di una tabella. Bisogna quindi stare attenti, quando si usa il descrittore `p`, per quale scopo si usa il comando `\\`; se serve per andare a capo dentro il capoverso si evita ogni fastidio se si usa al suo posto `\newline`, che non consente opzioni, ma che certamente non è ambiguo. Si osservi infine nella tabella 8.2 l'allineamento dei contenuti delle celle quando si usi il descrittore `p`.

8.4.2 Il raggruppamento delle celle

Le tabelle possono avere alcune celle adiacenti raggruppate a formare un'unica cella. LaTeX di per sé consente di raggruppare le celle solo orizzontalmente; mediante il pacchetto di estensione *`multirow`* è possibile raggrupparle anche verticalmente, ma il pacchetto è, per così dire, ancora in evoluzione, per cui

Opera	Brano
I promessi sposi	Quel ramo del lago di Como, che volge a mezzogiorno, tra due catene non interrotte di monti, tutto a seni e a golfi, a seconda dello sporgere e del rientrare di quelli, vien quasi a un tratto...

Tabella 8.2: Tabella la cui colonna di destra è specificata con `p{80mm}`

bisogna supplire a certe piccole manchevolezze con un po' di tentativi e di correzioni successive. In generale, però, una tabella ha un aspetto più professionale se non contiene celle raggruppate verticalmente.

Si usa il comando `\multicolumn` con la sintassi seguente per raggruppare alcune celle orizzontalmente:

`\multicolumn{`⟨*numero*⟩`}{`⟨*descrittore*⟩`}{`⟨*contenuto*⟩`}`

Il ⟨*numero*⟩ specifica quante celle bisogna raggruppare; il ⟨*descrittore*⟩ contiene i codici di allineamento e di separazione necessari per descrivere l'unica grande cella frutto del raggruppamento; il ⟨*contenuto*⟩ è il testo contenuto nell'unica grande cella.

Bisogna fare attenzione ad un dettaglio: i separatori di colonna da inserire nel ⟨*descrittore*⟩ riguardano solo il lato *destro* dell'unica cella, a meno che la cella non sia la prima della serie di celle in una riga: in questo unico caso bisogna esplicitare anche il descrittore da inserire nel lato sinistro della cella.

Per esempio, se si avesse una tabella in cui il campo dei descrittori fosse il seguente:

```
\begin{tabular}{|*6{c|}}
```

e si volessero raggruppare le celle 3, 4 e 5 in un'unica grande cella presente in una sola riga (per esempio la riga di intestazione), bisognerebbe specificare nella posizione della cella 3

```
\multicolumn3{c|}{Intestazione 3}
```

mentre se si volessero raggruppare le celle 1 e 2, sarebbe necessario specificare:

```
\multicolumn2{|c|}{Intestazione 2}
```

Per cui la prima riga della tabella, verosimilmente sarebbe descritta dal seguente codice:

```
\multicolum2{|c|}{Intestazione 2}
            &\multicolum3{c|}{Intestazione 3}&Intestazione 1\\
```

Nei precedenti esempi si noti che il numero di celle da raggruppare è formato da una sola cifra quindi non ha bisogno di essere racchiuso fra parentesi graffe;

questa è una regola generale; quando un argomento obbligatorio è formato da un solo oggetto (*token* nel gergo di TEX) non ha bisogno delle parentesi graffe, le quali, invece, sono necessarie ogni volta che l'argomento è costituito da diversi token; ovviamente non è vietato usare le graffe anche quando l'argomento è costituito da un solo token, ma omettendole si scrive un po' di meno e non si corre il rischio di dimenticare la graffa chiusa.

8.4.3 I separatori verticali

Per separare verticalmente le righe di celle generalmente non serve nulla; le tabelle sono più professionali se non contengono filetti o se ne contengono pochissimi.

In ogni caso LATEX consente di separare le celle con dei filetti o degli spazi aggiuntivi.

`\\` il comando `\\` serve per terminare una riga di celle; però esso consente anche di specificare uno spazio verticale da aggiungere alla normale spaziatura; si scrive allora, per esempio, `\\[1.5ex]` per terminare la riga e aggiungere uno spazio ulteriore di 1,5 ex fra il fondo della riga corrente e l'inizio della riga seguente. Si faccia attenzione che all'inizio della prima cella di ogni riga di qualunque tabella è inserito un oggetto invisibile che si chiama `\strut` (*pilastrino* in italiano); esso serve per garantire una altezza e una profondità minima ad ogni riga, in modo che le righe sembrino distanziate uniformemente indipendentemente dai loro ascendenti e discendenti; se come argomento facoltativo di `\\` si specifica uno spazio troppo piccolo, questo potrebbe risultare assorbito dalla profondità dello `\strut` della riga corrente.

`\tabularnewline` serve per terminare una riga della colonna e per passare alla riga successiva (vale anche per l'ambiente *array* da usarsi in ambiente matematico); sostanzialmente agisce come `\\`, ma si riferisce esclusivamente alla fine riga della tabella o della matrice; esso è utilissimo, per non dire indispensabile, quando l'ultima cella della riga è caratterizzata dal descrittore `p` (oppure con il pacchetto *array* anche con i descrittori `m` e `b`). Infatti in questi casi il comando `\\` è ambiguo: serve per andare a capo in una riga della cella oppure serve per andare a capo in una tabella? Se non si prendono particolari precauzioni il comando `\\` viene interpretato come fine riga del testo della cella, ma usando `\tabularnewline` si toglie ogni ambiguità e la tabella o la matrice viene composta correttamente.

`\hline` serve per tracciare un filetto orizzontale attraverso tutta la tabella; due comandi `\hline` di seguito l'uno all'altro producono due filetti distanziati di un paio di punti tipografici che rendono maggiormente evidente la separazione verticale. Il comando `\hline` deve essere specificato o prima della prima riga di celle, o dopo un comando `\\`.

`\cline` serve per tracciare un filetto orizzontale sotto *alcune* colonne adiacenti che devono venire specificate mediante la sintassi:

`\cline{`⟨*colonna iniziale*⟩`-`⟨*colonna finale*⟩`}`

Si noti che si possono ripetere due comandi `\cline` di seguito specificando le stesse colonne iniziale e finale ma non si ottiene un filetto doppio perché i due filetti si sovrappongono; si possono invece inserire due specificazioni con colonne diverse per mettere due filetti di seguito uno all'altro in orizzontale.

Esercizio 8.2 Si predisponga una tabella a più colonne e si inseriscano contenuti qualsiasi nelle celle; ma si specifichino tutti e tre i tipi di separatori verticali descritti in questo paragrafo, osservando bene che cosa succede quando lo spazio che costituisce l'argomento facoltativo del comando `\\` è di pochi punti. Ripetere la spaziatura con diversi valori di quello spazio.

8.4.4 Come rendere le tabelle un poco più aperte

Nonostante la presenza di uno `\strut` nella prima cella di ogni riga, talvolta sembra che questo `\strut` non sia abbastanza grande cosicché i filetti orizzontali sembrano toccare le lettere maiuscole o i discendenti delle minuscole.

Si può usare il parametro `\arraystretch` con la seguente sintassi:

`\renewcommand\arraystretch{`⟨*fattore*⟩`}`

Il ⟨*fattore*⟩ è un numero decimale, generalmente maggiore di uno, che rappresenta il fattore di scala con cui si vuol ingrandire verticalmente ogni spazio prodotto dagli `\strut`, e quindi con il quale si vuole ingrandire l'intera tabella. Per esempio si può dichiarare:

```
\begin{table}
\renewcommand\arraystretch{1.1}
\begin{tabular}{...}
...
\end{tabular}
\end{table}
```

In questo modo il valore unitario di default del parametro `\arraystretch` viene aumentato del 10% e tutte le righe della tabella risultano del 10% più alte e più profonde. Il fatto di cambiare il valore di `\arraystretch` all'interno di un ambiente, *table* in questo caso, confina la modifica all'interno dell'ambiente stesso.

Attenzione, così facendo si ingrandiscono verticalmente tutte le righe della tabella, anche quelle che non sono precedute o seguite da filetti orizzontali.

Può darsi che questo sia quello che si desidera, specialmente se si sta usando un fattore di scala non tanto superiore a uno. Tuttavia potrebbe essere una scelta non conveniente. Quando si vogliono ingrandire solo le celle precedute e/o seguite da filetti orizzontali non resta che inserire direttamente un altro `\strut` nelle righe che interessano. Il pilastrino che il compositore può usare si ottiene con il comando `\rule` e la sua sintassi è la seguente:

`\rule[`⟨*innalzamento*⟩`]{`⟨*base*⟩`}{`⟨*altezza*⟩`}`

Questo `\rule` definisce un riga verticale larga ⟨*base*⟩ e alta ⟨*altezza*⟩, innalzata rispetto alla linea di base del testo di ⟨*innalzamento*⟩. Per rendere questa riga invisibile basta specificarne la base nulla; per l'altezza si specifica quello che si vuole e si può innalzare o abbassare (innalzamento negativo) questa riga di quanto si vuole.

Personalmente io uso queste definizioni:

```
\begin{table}
\def\U{\rule{0pt}{3ex}}
\def\D{\rule[-0.5ex]{0pt}{0pt}}
\def\V{\U\D}
\begin{tabular}{...}
...
\end{tabular}
\caption{...}\label{...}
\end{table}
```

e inserisco `\U` nella prima cella di una riga *preceduta* da un filetto; inserisco `\D` nella prima cella di una riga *seguita* da un filetto; inserisco `\V` nella prima cella di una riga *preceduta e seguita* da un filetto.

Il comando `\def` appartiene al linguaggio base di TEX e sarebbe meglio non usarlo mai quando si scrive con LATEX. Infatti questo comando definisce una nuova istruzione e, per sbaglio, si potrebbe ridefinire una istruzione interna di LATEX perché non viene eseguita nessuna verifica della liceità di questa definizione. Tuttavia in questo caso le tre istruzioni `\U`, `\D` e `\V` non corrispondono a nessun'altra definizione interna a LATEX, ma, quello che più conta, la definizione mediante `\def` rimane valida solo all'interno dell'ambiente *table*, per cui all'uscita dall'ambiente, dopo `\end{table}`, quei comandi riprendono il significato che avevano prima di entrare nell'ambiente *table*.

Così ho fatto in tutte le tabelle esposte finora e così farò in tutte le prossime tabelle.

Esiste la possibilità di usare il pacchetto *booktabs*; questo contiene le definizioni per i tre filetti `\toprule`, `\midrule` e `\bottomrule` da mettere rispettivamente sopra la riga di intestazione, sotto la riga di intestazione e sotto l'ultima riga della tabella. Il primo e l'ultimo sono leggermente più spessi del filetto intermedio. Ovviamente tutti i parametri stilistici sono configurabili. Questi filetti assicurano uno spazio adeguato in modo che risultino sufficientemente distanziati dal contenuto delle celle adiacenti. Sono incompatibili con i filetti verticali, quindi 'obbligano' a comporre le tabelle in modo professionale, perciò questo pacchetto viene sempre fortemente consigliato. Però bisogna ricordare che questi filetti danno dei piccoli problemi se si decide di mettere uno sfondo colorato alla tabella, a singole celle, o a un'intera colonna o un'intera riga.

Per evitare questi piccoli problemi bisogna ricorrere ai pilastrini esposti sopra, anche se questo richiede una maggiore attenzione da parte del compositore/

tastierista per non dimenticare nessun elemento compositivo da introdurre a mano.

8.5 Le tabelle di larghezza specificata

La tabella generata con l'ambiente asteriscato prevede che sia di larghezza specificata proprio mediante l'argomento ⟨*larghezza*⟩.

Per ottenere quella larghezza è necessario che le singole celle possano allargarsi quanto basta perché la tabella complessiva raggiunga il valore desiderato.

A questo scopo LaTeX dispone di una specificazione di una lunghezza 'elastica', detta anche 'gomma', che internamente è caratterizzata da tre valori, la lunghezza naturale, l'allungamento e l'accorciamento; se l'allungamento e l'accorciamento corrispondono a valori nulli, essa non si distingue da una lunghezza rigida, che è un altro tipo di lunghezza usato internamente da LaTeX.

Se invece l'allungamento e o l'accorciamento corrispondono a valori 'infiniti', si è in presenza di una 'gomma infinitamente allungabile o accorciabile'.

Nel caso delle tabelle si predispone allora una *@-espressione* che al suo interno, oltre agli altri eventuali elementi contenga anche la seguente espressione

```
\extracolsep{\fill}
```

Questa espressione inserisce una lunghezza infinitamente allungabile *in tutti i successivi* separatori di colonna, e questa gomma appare a destra degli altri separatori già presenti; ad esempio:

```
\begin{tabular*}{\textwidth}%
        {@{\extracolsep{\fill}\vline\hspace{\tabcolsep}}l|*3{c|}}
```

permette di definire una tabella con quattro colonne, delle quali la prima abbia tutte le celle con il loro contenuto allineato a sinistra e le altre tre abbiano le celle con il loro contenuto centrato; le quattro celle di ogni riga, quindi le quattro colonne, sono contornate da filetti verticali; la prima cella, inclusi i separatori, ha le sue dimensioni naturali, mentre le celle successive hanno lo spazio a sinistra di ogni contenuto allargato di quanto basta per far sì che l'intera tabella sia larga quanto specificato.

Purtroppo questo modo di procedere va abbastanza bene per tabelle che quasi raggiungono naturalmente la larghezza desiderata, cosicché lo spazio che è necessario aggiungere sia otticamente trascurabile rispetto alla spaziatura normale. Invece se le celle hanno un contenuto abbastanza corto e la tabella deve essere allargata molto, il risultato è tutt'altro che ottimale. Si vedano le tabelle 8.3 e 8.4 per rendersi conto di quanto affermato.

Ci sono diverse soluzioni a questo problema; una consiste nel richiamare il pacchetto di estensione `tabularx`, alla cui documentazione si rinvia il lettore.

Un'altra soluzione consiste nel definire un nuovo ambiente che calcoli la larghezza naturale delle singole celle e distribuisca omogeneamente lo spazio

Famiglia			
Rossi Mastrodomenico Papadopoulos	Mario Giovanni Battista Filolaos	Maria Elisabetta Marta Penelopi	Filippo Elena Beatrice Irini

Tabella 8.3: Tabella di larghezza pari alla giustezza del testo ottenuta allargando poco le singole celle

Cognome	Età	Peso	Altezza
Rossi	50	85	182
Mastrodomenico	65	72	175
Papadopoulos	47	75	160

Tabella 8.4: Tabella di larghezza pari alla giustezza del testo ottenuta allargando troppo le singole celle

mancante a sinistra e a destra di ogni separatore di colonna, in modo che lo spazio bianco necessario per ottenere il risultato desiderato sia distribuito più uniformemente. Nella tabella 8.4 si osserverebbe allora che le colonne 2, 3 e 4, avrebbero davvero il loro materiale centrato, ma anche la colonna 1 avrebbe margini un poco più ampi così da ridurre lo spazio bianco nelle colonne di destra; si osservi il risultato nella tabella 8.5.

Cognome	Età	Peso	Altezza
Rossi	50	85	182
Mastrodomenico	65	72	175
Papadopoulos	47	75	160

Tabella 8.5: Tabella di larghezza pari alla giustezza del testo ottenuta allargando lo spazio fra le singole celle

In ogni caso ci si ricordi che la presenza dei filetti verticali peggiora solamente l'effetto visivo, se bisogna davvero forzare un poco la composizione; per cui è quanto mai opportuno cercare di evitare l'inserimento dei filetti, in primis quelli verticali, possibilmente anche quelli orizzontali, accettando solo quelli che delimitano la tabella e il filetto sotto la prima riga che contiene le intestazioni delle colonne. Questa è la raccomandazione stilistica contenuta in moltissimi manuali di tipografia; è giusto che LaTeX consenta di inserire tutti i filetti che si vogliono, ma sta al compositore scegliere se metterli, quali mettere, e dove metterli.

Confesso: ho barato un pochino dicendo che basta usare un ambiente che

calcoli la larghezza naturale delle singole celle e distribuisca omogeneamente lo spazio mancante a sinistra e a destra di ogni separatore di colonna. È vero; altrove ho generato e usato un simile ambiente (pacchetto *widetable*), ma nell'esempio della tabella 8.5 ho provato alcune volte modificando ad occhio il valore dello spazio di separazione intercolonna; mi sono bastati quattro o cinque tentativi per arrivare a definire che detto spazio, `\tabcolsep`, deve valere 8,9 mm; in questo modo, senza ricorrere a nessuna acrobazia di programmazione in linguaggio TeX; la tabella 8.5 è stata composta con i comandi

```
\begin{table}[ht]
\def\V{\rule[-1ex]{0pt}{3.5ex}}
\def\S{\rule{0pt}{3,5ex}}
\setlength{\tabcolsep}{8.9mm}%  <---- spazio intercolonna
\begin{tabular}{|l|*3{c|}}
\hline
Cognome\V  & Età   &Peso  & Altezza  \\
\hline
Rossi\S    & 50    & 85   & 182      \\
...
\end{tabular}
\end{table}
```

Si noti che non si è usato l'ambiente *tabular**, ma il semplice ambiente *tabular*; la larghezza è stata ottenuta modificando lo spazio intercolonna finché l'occhio ha confermato che la tabella era larga quanto la giustezza.

Vale la pena di citare il pacchetto *tabu* alla cui documentazione si rimanda il lettore; questo pacchetto consente di gestire i colori sia dei filetti, sia del testo, sia dello sfondo delle celle; consente di agire come e meglio di *tabularx* per generare una tabella di larghezza specificata, ma consente anche di allargare una data tabella di un ammontare specificato; consente l'uso delle celle di tipo `X`, tipiche dell'ambiente *tabularx* che le definisce, anche nella versione "long", cioè sia nell'ambiente *tabu* sia nell'ambiente *longtabu*; se in *tabu* o in *longtabu* si usano celle con i descrittori X, `p`, `m`, `b` il comando `\\` serve per terminare la riga della tabella, senza bisogno di ricorrere a `\tabularnewline`; consente l'incolonnamento basato sul separatore decimale, l'uso delle colonne `S` o `s` definite dal pacchetto *siunitx*; consente di colorare le righe di una tabella secondo colori definiti con i comandi del pacchetto *xcolor*, ma con comandi suoi propri che regolano meglio il riempimento dello sfondo anche in presenza dei filetti definiti con il pacchetto *booktabs*; insomma consente di fare molte delle cose che si potrebbero ottenere caricando diversi pacchetti (che il pacchetto *tabu* stesso emula o che vengono caricati dall'utente) ma consente anche di fare cose nuove che con quei pacchetti non si possono ottenere. Merita ricordare che l'ambiente *tabu* può essere annidato dentro *tabu* stesso, *tabular*, *tabular** e *tabularx* e viceversa; con alcune limitazioni lo si può annidare anche dentro le tabelle lunghe, ma è meglio riferirsi alla documentazione per valutare con attenzione queste limitazioni. L'ambiente *tabu* si può usare anche in modo matematico e può sostituire l'ambiente *array*. Nel

capitolo 14 si mostra un esempio il cui risultato non può essere ottenuto con gli altri ambienti e pacchetti LaTeX.

8.6 Problemi compositivi delle tabelle

Le tabelle sono difficili da comporre, non tanto perché i comandi necessari siano complessi, quanto perché esse devono venire progettate con anticipo, non devono essere composte sperando che vada tutto bene.

Ma anche con una buona impostazione il contenuto delle celle richiede aggiustamenti che non possono essere previsti in automatico dal programma, ma devono essere valutati con attenzione dal compositore.

Ecco quindi che nei prossimi paragrafi si daranno indicazioni per rimediare ad alcuni problemi che talvolta si presentano; questi suggerimenti, però, servano anche come stimolo per trovare autonomamente le soluzioni più consone ad altri problemi specifici che potrebbero manifestarsi.

8.6.1 Tabelle troppo larghe

Può darsi che una tabella sia così larga da fuoriuscire dalla giustezza; ma in questo caso si può rimediare in diversi modi a seconda che l'eccesso di larghezza rispetto alla giustezza sia di pochi punti, di pochi millimetri, o invece sia abbastanza consistente.

Nei primi due casi si possono mettere due *@-espressioni* 'vuote' come delimitatori a sinistra della prima cella e a destra dell'ultima cella; così facendo si eliminano le sporgenze dei filetti (se ci sono) fuori della giustezza della tabella.

Se ciò non bastasse, si potrebbe ancora ridurre lo spazio fra le colonne; questo spazio è dato da `\tabcolsep` che di default vale 6 pt. Si noti: questo spazio è quello che viene messo a sinistra e, rispettivamente, a destra di ogni separatore verticale; quindi i contenuti delle celle sono separati dal doppio, vale a dire da 12 pt. Non si perde molto nella composizione della tabella se si imposta il valore di `\tabcolsep` ad un valore di 3 o 4 pt; a seconda del numero delle colonne si possono recuperare diversi millimetri di larghezza.

Un'altra soluzione può consistere nel ridurre il corpo dei font con cui è composta la tabella; passando a `\small` o a `\footnotesize` spesso si risolvono problemi importanti. Basta procedere così:

```
\begin{table}
\small % oppure \footnotesize
\begin{tabular}{...}
...
\end{tabular}
\caption{...}
\label{tab:...}
\end{table}
```

Se il problema non è molto grave, sarebbe meglio evitare di scendere al corpo corrispondente a `\footnotesize`.

Simile a questa soluzione, esiste la possibilità di ricorrere al pacchetto *graphicx* che fornisce il comando `\resizebox`; questa soluzione è più flessibile, perché consente di scegliere il fattore di scala praticamente con continuità. Il suo unico difetto è che si può usare questa soluzione solo quando il file finale deve essere in formato PostScript oppure PDF, perché solo questi formati sono capaci di scalare con fattori di scala qualunque gli oggetti che contengono. Il file DVI, invece, non è in grado di scalare gli oggetti che contiene.

Resta il terzo caso dove la larghezza della tabella, nonostante le cure descritte nei capoversi precedenti, continua ad eccedere la giustezza del testo; se si tratta di una decina di millimetri al massimo, secondo il giudizio estetico del compositore, la si può collocare centrata lasciandola sporgere metà a sinistra e metà a destra. L'operazione è molto semplice, perché basta racchiudere tutta la tabella dentro una scatola di larghezza specificata; il comando `\makebox` che esegue questa operazione agisce secondo la seguente sintassi:

`\makebox[`⟨*larghezza*⟩`][`⟨*collocazione*⟩`]{`⟨*testo*⟩`}`

dove ⟨*larghezza*⟩ rappresenta la larghezza vera o apparente della scatola così creata; la ⟨*collocazione*⟩ specifica se il `{`⟨*testo*⟩`}` va collocato al centro, oppure accostato a sinistra o a destra dentro la scatola. Il ⟨*testo*⟩ può essere effettivamente del testo comune, ma può anche essere una tabella o qualunque altro oggetto già composto e trattato come un solo oggetto; anche una figura o un diagramma può essere usato come ⟨*testo*⟩. La ⟨*larghezza*⟩ è una larghezza vera se il ⟨*testo*⟩ ha una larghezza nominale inferiore a quella specificata, mentre essa è una larghezza apparente nel caso opposto; in altre parole se il ⟨*testo*⟩ è più largo di quanto specificato, lo si fa apparire largo esattamente come ⟨*larghezza*⟩.

Per centrare una tabella un pochino più larga della giustezza corrente basta dunque inserirla dentro una scatola larga quanto `\linewidth` specificandone la posizione al centro della scatola (cioè non specificando nessuna ⟨*collocazione*⟩ visto che la posizione centrata è quella di default). Un costrutto del tipo

```
\begin{table}
\makebox[\linewidth]{%       Inizio scatola
   \begin{tabular}{...}
   ...
   \end{tabular}%
}%                           Fine scatola
\caption{...}\label{tab:...}
\end{table}
```

risolve il problema.

Tuttavia può ancora succedere che la tabella non possa essere 'aggiustata' con nessuno dei trucchi sopra esposti. Allora può succedere che l'eccesso di larghezza rispetto alla giustezza sia ancora dominabile con una rotazione di 90° in senso

antiorario. Attenzione: la rotazione deve sempre avvenire in senso antiorario, indipendentemente dal fatto che la tabella venga poi fatta flottare in un recto invece che in un verso. Questo implica che per leggere la tabella, il documento debba venire ruotato sempre in verso *orario* di 90°.

Per eseguire questa rotazione, tenendo conto che la rotazione coinvolge anche la didascalia, è opportuno usare il pacchetto di estensione *rotating*, e al suo ambiente *sidewaystable*, alla cui documentazione si rinvia per reperire le necessarie informazioni d'uso.

8.7 Tabelle troppo lunghe

Con 'tabella lunga' si intende una tabella che non stia in una pagina. In questo caso ci sono diverse soluzioni, la prima delle quali è domandarsi se la tabella sia stata concepita bene; potrebbe darsi che scambiando le righe con le colonne la tabella diventi più larga e meno lunga; vale quindi la pena di ripensare alla possibilità di sistemare la tabella con questo semplice artificio.

Se la tabella dovesse risultare troppo larga, ma tale da poter stare in una sola pagina, si possono adottare i rimedi esposti nel paragrafo precedente.

Ma se la tabella è veramente lunga bisogna ricorrere a pacchetti di estensione, i più noti dei quali sono *longtable* e *supertabular* alla cui documentazione si rinvia, visto che questo tipo di composizione richiede un approccio abbastanza delicato.

Concettualmente il problema è semplice: si tratta di spezzare una lunga tabella in monconi tali da poter essere sistemati ciascuno in una pagina sola; per ovvie ragioni estetiche le colonne devono mantenersi della stessa larghezza in tutti i monconi, ma bisogna anche ripetere le intestazioni delle colonne, per non dover sfogliare pagine indietro per sapere se nella terza colonna compare la densità relativa oppure la massa volumica delle sostanze elencate nella tabella.

È opportuno anche che in ogni moncone della tabella che prosegue nella pagina successiva sia inserita una specie di intestazione di piè di pagina che specifica che la tabella continua; analogamente in ogni moncone della tabella, eccetto il primo, conviene mettere una intestazione dalla quale si capisca che si è nel cuore della tabella, non all'inizio né alla fine; questo può ottenersi anche ripetendo la didascalia alla fine della quale sia scritto 'continua', possibilmente fra parentesi e in corsivo, così da separare bene questa informazione dal resto della didascalia. Ovviamente il contatore della tabella non deve aumentare di una unità ad ogni moncone.

Quando si sono ben afferrati questi concetti, abbastanza semplici, allora diventa abbastanza facile comprendere la documentazione dei due pacchetti sopra indicati per potersi regolare al meglio e comporre una tabella professionale.

Un suggerimento non guasta. Conviene inserire il codice della lunga tabella in un file TEX a parte; solo il codice, senza `\documentclass` e senza l'ambiente *document*. Conviene fare uso del pacchetto *afterpage* e servirsi del comando `\afterpage`. Infatti, benché una lunga tabella non sia mobile, è opportuno che

il primo moncone della tabella cominci all'inizio di una nuova pagina; mediante `\afterpage`, pertanto, diventa quasi spontaneo scegliere una posizione adeguata per dare il comando di leggere e di assorbire, usando il comando `\input`, il file che contiene il codice per la compilazione della tabella; questa automaticamente comincia all'inizio della prima pagina disponibile, finisce qualche pagina dopo, e il testo ricomincia subito dopo. Come si capisce bene, è estremamente importante scegliere il punto giusto per l'inserimento della lunga tabella; non necessariamente questo punto corrisponde alla fine di un capitolo; dipende dal genere di testo che si sta componendo. Se il codice per la lunga tabella è contenuto nel file `tabellalunga.tex`, memorizzato nella cartella `./file-inclusi`, allora quanto detto sopra corrisponde semplicemente a dare il comando

```
\afterpage{\input{file-inclusi/tabellalunga}}
```

Nel capitolo 24 si è fatto appunto così.

Si noti ancora che con le tabelle che si sviluppano su più pagine è necessario mettere la didascalia prima della tabella; per questo, a differenza dell'uso americano, rispecchiato in questo libro composto con la classe *book* alla quale non si sono apportate modifiche, sarebbe opportuno mettere sempre la didascalia prima di qualsiasi tabella, non solo prima di quelle lunghe. Per questo scopo, al fine di non dover aggiustare le cose a mano per ogni tabella che si compone, è possibile avvalersi del pacchetto *topcapt* che mette a disposizione del compositore il comando `\topcaption`, da usare esattamente come `\caption`; esso usa le spaziature giuste per mettere la didascalia sopra, invece che sotto, un qualunque oggetto flottante; si userà quindi `\topcaption` per le didascalie delle tabelle e `\caption` per le didascalie delle figure.

Nel prossimo paragrafo sarà mostrato un esempio d'uso delle potenzialità dei pacchetti di estensione per le tabelle, descrivendo quanto si è fatto per comporre le lunghe tabelle del capitolo 24.

8.8 Pacchetti di estensione per le tabelle

Oltre ai già citati pacchetti *tabularx*, *tabu*, *longtable* e *supertabular*, conviene riprendere il discorso del pacchetto *array* appena menzionato in relazione all'incolonnamento basato sul separatore decimale. Per dirla tutta, questo lavoro viene eseguito da una piccola estensione del pacchetto *array* che ne sfrutta appieno le potenzialità; si tratta del pacchetto *dcolumn* che è in dotazione standard di ogni distribuzione del sistema TeX.

Questo pacchetto definisce una colonna di nuovo tipo e con il codice `D`; ma accetta diverse personalizzazioni che è meglio esaminare nella documentazione.

Qui si vuole richiamare l'attenzione del lettore sul fatto che il pacchetto *array* mette a disposizione del compositore due nuovi descrittori di colonna simili a `p{...}`; si tratta di `m{...}` e `b{...}`. Agiscono nello stesso modo, nel senso che tutti e tre i descrittori richiedono una larghezza di colonna al posto dei puntini; la differenza è che il contenuto della scatola di testo prodotto con `p{...}` ha la

sua prima riga allineata con la prima o unica riga delle celle adiacenti; `b{...}` ha invece la sua ultima riga allineata con le celle adiacenti; infine `m{...}` è centrato con la sua riga mediana rispetto alle celle adiacenti.

Esercizio 8.3 Che cosa succederebbe alla tabella 8.2 se si usasse il descrittore `m{80mm}` oppure il descrittore `b{80mm}` al posto del descrittore `p{80mm}`?

Questi tre descrittori sono utilissimi, in particolare lo scrivente ha notato che, secondo i suoi gusti e il tipo di tabelle che compone, le celle con il blocco di testo centrato verticalmente sono più gradevoli rispetto a quando sono allineate sulla base della prima o dell'ultima riga. Ripeto, si tratta di una questione di gusti e, principalmente, del tipo di tabelle che si desidera comporre; in ogni caso è opportuno poter disporre di una scelta.

È molto utile che il pacchetto *array* definisca il comando `\arraybackslash`; se l'ultima colonna ha il descrittore che esplicitamente o implicitamente contiene una delle dichiarazioni `\centering`, `\raggedright` oppure `\raggedleft`, l'ambiente di tabulazione non sa se un comando `\\` serva per andare a capo dentro la cella o se serva per terminare la composizione di una riga di celle; questo è particolarmente importante se si fa uso delle dichiarazioni di colonna introdotte con la sintassi descritta qui di seguito. Ecco allora che la dichiarazione `\arraybackslash` introdotta, per esempio, dopo `\centering`, permette di stabilire che il prossimo comando `\\` indica la terminazione della composizione della riga di celle. Con l'ambiente *tabu* questi accorgimenti sono superflui.

Il pacchetto *array* consente anche di comporre tabelle miste, in cui alcune colonne sono testuali ed altre puramente numeriche; anzi la caratteristica di *array* è quella di inserire una dichiarazione all'inizio di tutte le celle della stessa colonna e di porre una corrispondente dichiarazione alla fine di ciascuna di queste celle.

I descrittori delle colonne possono diventare particolarmente complessi, ma se si fa attenzione a descrivere con ordine ciascuna colonna, la cosa non è affatto complicata.

I nuovi 'comandi' per inserire queste dichiarazioni iniziali e finali in tutte le celle di una stessa colonna sono `>` e `<`. Essi seguono la sintassi seguente:

`>{`⟨*dichiarazioni iniziali*⟩`}`⟨*descrittore di colonna*⟩`<{`⟨*dichiarazioni finali*⟩`}`

Val più un esempio per descriverne l'azione, che non tante parole[1]; predisponiamo una piccola tabella con le colonne allineate a sinistra, al centro, e a destra, ma dove la prima colonna deve avere tutti gli elementi in grassetto, la seconda deve contenere una descrizione in corsivo, la terza deve contenere un numero che rappresenta un prezzo e quindi deve essere presente l'unità monetaria; la tabella 8.6 è stata composta con i seguenti comandi

[1] In questo stesso paragrafo si vedrà più avanti come è stata composta la tabella 24.8 facendo uso di questi descrittori speciali.

Prezziario		
Pantaloni	*alla zuava*	35,00 €
Pantaloni	*corti*	12,50 €
Camicie	*a scacchi*	22,75 €
Camicie	*botton down*	31,20 €
Calze	*di lana*	8,33 €
Calze	*di cotone*	5,00 €
Calze	*collant*	15,70 €

Tabella 8.6: Tabella composta con le estensioni del pacchetto *array*

```
\begin{table}\centering
\begin{tabular}{>{\bfseries}l>{\itshape}cr<{\enspace\texteuro}}
\multicolumn3c{\textsf{\bfseries Prezziario}}\\\hline
Pantaloni   & alla zuava     & 35,00\rule{0pt}{2.5ex}\\
Pantaloni   & corti          & 12,50 \\
Camicie     & a scacchi      & 22,75 \\
Camicie     & botton down    & 31,20 \\
Calze       & di lana        &  8,33 \\
Calze       & di cotone      &  5,00 \\
Calze       & collant        & 15,70
\end{tabular}
\caption{...}\label{tab:arraypack}
\end{table}
```

e si vede quanto sia conveniente l'uso dei descrittori iniziali e di quelli finali per la composizione di colonne omogenee.

Ma una delle cose più utili del pacchetto *array* è costituito dalla possibilità di definire nuovi descrittori di colonna che possono ulteriormente semplificare la composizione delle tabelle.

Infatti il pacchetto *array* mette a disposizione il comando `\newcolumntype` con la seguente sintassi:

`\newcolumntype{`⟨*descrittore*⟩`}{`⟨*specifiche*⟩`}`

Il ⟨*descrittore*⟩ è formato da una sola lettera minuscola o maiuscola che non sia già stata usata per i descrittori esistenti e le ⟨*specifiche*⟩ sono costituite da un descrittore completo delle sue specificazioni particolari descritte mediante i comandi `>{...}` e/o `<{...}`; estendendo il concetto, il ⟨*descrittore*⟩ può corrispondere a delle ⟨*specifiche*⟩ che fanno uso di più descrittori.

Per esempio, nella tabella 8.6 sono stati usati per le tre colonne i descrittori

```
>{\bfseries}l>{\itshape}cr<{\enspace\texteuro}
```

Facendo uso del comando appena descritto si sarebbero potuti definire tre nuovi descrittori:

```
\newcolumntype{L}{>{\bfseries{l}}}
\newcolumntype{C}{>{\itshape{c}}}
\newcolumntype{E}{r<{\enspace\texteuro}}
```

e poi si sarebbe potuto descrivere il comando di apertura della tabella con:

```
\begin{tabular}{LCE}
```

È chiaro che se questi tipi di colonne vengono usati spesso, è conveniente eseguire le definizioni dei nuovi tipi di colonna una volta per tutte nel preambolo e poi usarli regolarmente nelle varie tabelle che si intendono comporre.

Al limite, se diverse tabelle usano la stessa sequenza di descrittori di colonna, questa può essere inclusa in un unico descrittore; per esempio

```
\newcolumntype{T}{LCE}
...
\begin{tabular}{T}
```

Anche nelle lunghe tabelle del capitolo 24 si è fatto nello stesso modo.

Per comodità del lettore, si riporta qui l'insieme delle dichiarazioni usate per comporre la tabella 24.8, compreso l'inizio e la fine della tabella, ma senza il contenuto, per ovvi motivi di brevità.

```
\newcommand*{\hstrut}{\rule{0pt}{3ex}\hskip 0sp}
\newcolumntype{B}{>{\bfseries}c}
\newcolumntype{M}{>{$\displaystyle}c<{$}}
\newcolumntype{R}{>{\raggedright}p{.35\textwidth}}
\newcolumntype{S}{>{\hstrut}p{.45\textwidth}<{\vspace*{1ex}}}
\newlength{\stretchandshrink}
\setlength{\stretchandshrink}{0pt plus 1fil minus 1fil}
...
\begingroup
\setlength\LTleft{0pt}
\setlength\LTright{0pt}
\begin{longtable}{@{\extracolsep{\stretchandshrink}}MRS@{}}%
    \caption{Simboli matematici\label{tmat}}\\
    \noalign{\smallskip\hrule\medskip}
    \multicolumn1B{Simbolo}&
    \multicolumn1B{Significato}&
    \multicolumn1B{Note}\\
    \noalign{\smallskip\hrule\medskip}
\endfirsthead
    \multicolumn3l{\emph{Continua tabella \thetable}}\\
    \noalign{\smallskip\hrule\medskip}
    \multicolumn1B{Simbolo}&
    \multicolumn1B{Significato}&
    \multicolumn1B{Note}\\
```

```
    \noalign{\smallskip\hrule\medskip}
\endhead
    \noalign{\smallskip\hrule\smallskip}
    \multicolumn3r{\emph{continua}}
\endfoot
    \noalign{\smallskip\hrule}
\endlastfoot
... & ... & ... \\% qui comincia la tabella vera e propria
\end{longtable}
```

Come si vede `\caption` permette di comporre la didascalia della lunga tabella al punto giusto, prima della tabella. `\endfirsthead` termina la descrizione necessaria per comporre le intestazioni delle colonne nel primo moncone. `\endhead` termina la descrizione necessaria per comporre l'intestazione delle colonne nei monconi successivi al primo; nello stesso tempo esso contiene anche l'informazione che la tabella che compare nella pagina corrente è una continuazione di una tabella iniziata in pagine precedenti e il riferimento `\thetable` serve appunto per ripetere il numero della tabella senza incrementarne il contatore. `\endfoot` termina la descrizione necessaria per comporre la fine di ogni moncone tranne l'ultimo. `\endlastfoot` termina la descrizione necessaria per comporre la fine dell'ultimo moncone. I comandi per queste composizioni vengono memorizzati ed eseguiti ogni volta che ce ne sia la necessità, perché l'ambiente *longtable* provvede da solo a spezzare la tabella dove è necessario.

Fra i comandi usati si notano alcuni comandi primitivi, come `\noalign` che serve per inserire qualcosa dentro ad una tabella, senza tenere conto dei descrittori delle colonne; `\hrule` indica un filetto orizzontale che attraversa tutta la giustezza dello specchio di stampa; `\medskip` indica di inserire uno spazio verticale medio (circa mezza riga); `\smallskip` indica di inserire uno spazio verticale piccolo (circa di un quarto di riga). `\emph` serve per evidenziare una parola o una breve locuzione, nel senso che, se essa va evidenziata all'interno di un brano composto in tondo, esso evidenzia il suo argomento componendolo in corsivo; al contrario esso evidenzia in tondo all'interno di un brano in corsivo. La prima colonna è composta in modo matematico e in stile `\displaystyle`; la seconda colonna è composta in bandiera allineata a sinistra con una giustezza di 0,35 volte la giustezza della pagina; la terza colonna è composta con una giustezza pari a 0,45 volte la giustezza della pagina, ma solo dopo aver inserito un pilastrino definito come uno strut alto 3 ex seguito da uno spazio nullo[2]; questo spazio nullo rappresenta un trucchetto per assicurare che quanto segue sia eventualmente divisibile in sillabe (vedi capitolo 25). Il comando `\thetable` serve per scrivere il contenuto del contatore `table`.

Va ancora commentata l'impostazione delle due lunghezze `\LTleft` a sinistra della tabella e `\LTright` alla sua destra; esse sono collocate all'esterno della

[2] L'unità di misura `sp` indica la frazione $1/2^{16}$ di un punto tipografico; la notazione in base 2 è chiarissima per un calcolatore, mentre la notazione decimale 1/65536 non è altrettanto significativa.

tabella, e aggiustandone i valori è possibile o lasciare alla tabella la sua larghezza naturale (basta che l'una e/o l'altra rappresentino lunghezze elastiche di allungabilità "infinita") sia, impostandone valori rigidi, allargare o restringere un pochino la tabella purché almeno una colonna sia allargabile o accorciabile a piacere; è quello che si è fatto nelle due lunghe tabelle del capitolo 24, ed è per questo che il primo separatore delle colonne è indicato con una *@-espressione* che contiene come argomento del comando `\extracolsep` una larghezza elastica `\stretchandshrink` di valore nominale nullo ma di allungabilità e di accorciabilità infinite (di primo ordine); perché questo? Perché in questo modo la tabella può *allargarsi o restringersi* a piacere.

L'ambiente *longtable* scrive nel file ausiliario le larghezze delle colonne di ogni moncone e richiede di comporre la tabella almeno tre volte per essere sicuri che tutti i monconi abbiano le colonne con le stesse larghezze; siccome ogni documento viene composto sempre diverse volte, se non altro per controllare le versioni in bozza prima di comporre la versione finale, questa necessità di comporre la tabella diverse volte in realtà è un prezzo piccolissimo da pagare per ottenere il risultato voluto. Inoltre è bene sapere che *longtable* non inizia mai una pagina nuova se non alla fine di una riga della tabella, quindi non può succedere che il testo di una cella risulti diviso fra due pagine consecutive.

Nel paragrafo 19.9.3 sono esposti alcuni altri argomenti particolari relativi alle tabelle e alle figure inserite in alcune celle. Vi si definiscono dei nuovi comandi, ed è per questo che l'argomento è rinviato a quel paragrafo.

Nonostante il LaTeX3 Team abbia già predisposto i diversi pacchetti di estensione, descritti sopra, che consentono di migliorare la composizione delle tabelle e delle matrici, è opportuno segnalare una piccola lista, che non esaurisce tutte le possibilità disponibili negli archivi internazionali, ma che vale la pena di tenere presente in ogni caso.

Innanzi tutto i pacchetti *amsmath* e soci, prodotti dalla American Mathematical Society, estendono complessivamente le varie maniere di incolonnare oggetti, con riferimento specifico alle equazioni o a espressioni matematiche di vario genere; per questo pacchetto si rinvia la descrizione al capitolo 14.

L'Institution of Electrical and Electronics Engineers ha prodotto le sue classi per comporre gli articoli da pubblicare nelle sue riviste; ma ha messo a disposizione dei suoi autori anche un pacchetto separato *IEEEtrantools* che contiene diversi comandi e ambienti speciali, fra i quali gli ambienti per costruire tabelle e matrici in modo molto avanzato e non vincolato in modo così stretto alla sintassi tradizionale di LaTeX; questa infatti serve per 'mascherare' la complicazione della creazione della *riga modello* necessaria al comando primitivo `\halign` che è quello che effettivamente esegue poi la composizione delle tabelle e delle matrici. La documentazione si legge con `texdoc IEEEtrantools`, ma è preferibile leggerne le specifiche e gli esempi d'uso con `texdoc IEEEtran`.

Un altro pacchetto molto interessante è quello scritto (insieme ad altri utili pacchetti) da Mark D. Wooding, *mdwtab* con il suo corollario *mathenv*, che permettono di fare più o meno le stesse cose che si possono avere usando assieme i pacchetti *amsmath* e *array*. Alcune cose possono portare a risultati migliori

che non con quei pacchetti, conservando però una grande versatilità e omogeneità di comportamento, in modo che con un unico ambiente per le tabelle si possono comporre sia tabelle vere e proprie, sia matrici matematiche, sia incolonnamenti di colonne miste; per le matrici vere e proprie sono disponibili ambienti appositi, in un certo senso più versatili di quelli, ottimi, messi a disposizione dal pacchetto `amsmath`.

Allo stesso tempo si possono garantire spaziature migliori sopra e sotto i filetti in modo che diventa veramente facile produrre tabelle o matrici con filetti attorno a tutte o a molte caselle con le spaziature giuste e senza che i filetti si interrompano o che i 'pilastrini' siano inefficaci. È vero che i filetti non dovrebbero essere usati per niente, eccettuati quelli che delimitano superiormente e inferiormente una tabella e il filetto che stacca le celle di intestazione dalle altre; tuttavia talvolta è necessario produrre tali tabelle, magari per replicare tabelle altrui anche nello stile compositivo, e i pacchetti di Wooding sono molto versatili.

Va notato, comunque, che per tabelle, matrici e sistemi di equazioni Wooding ridefinisce i comandi ordinari di LaTeX così che in effetti non c'è da documentarsi troppo, se non per imparare le infinite varianti e possibilità offerte dai suoi pacchetti.

Del pacchetto `tabu` e del sua ambiente *`tabu`* si è già detto in precedenza.

Capitolo 9

LaTeX: figure

9.1 Le figure e le immagini

Con LaTeX è opportuno distinguere bene fra 'figura' e 'immagine' o 'disegno' o 'grafico'; la *figura* è un oggetto non verbale (fotografia, disegno, grafico, schema, diagramma, eccetera) dotato di un numero e di una didascalia che LaTeX deve sistemare in qualche posto dove ci sia abbastanza spazio, visto che generalmente l'oggetto è di dimensioni relativamente grandi.

Invece l'*oggetto grafico* che viene trattato da LaTeX come una figura può avere le forme più disparate e può avere anche le origini più diverse. Potrebbe essere prodotto con una macchina fotografica, con un software da disegno, con i comandi stessi di LaTeX e dei suoi pacchetti di estensione; potrebbe anche essere costituito da codice di programmazione che verrà tradotto in grafico dal driver che rende leggibile agli umani il documento.

In questo capitolo si parlerà solo dell'ambiente *figure* e dei modi di produrre materiale grafico con LaTeX stesso, accennando appena ai disegni eseguibili con i pacchetti di estensione.

9.2 L'ambiente `figure`

L'ambiente *figure* è del tutto analogo all'ambiente *table*; serve per rendere flottante un oggetto e per dargli una didascalia il cui primo termine è il nome 'Figura', o 'Figure', o 'Abbildung' o ..., a seconda della lingua in uso, seguito dal suo numero progressivo e dalla didascalia vera e propria, composta come sempre da un titolo e da una spiegazione (facoltativa).

La sintassi è simile a quella dell'ambiente *table*:

```
\begin{figure}[⟨posizione⟩]
...
\end{figure}
```

e i parametri di ⟨*posizione*⟩ `h`, `t`, `b`, `p` e `!` sono gli stessi dell'ambiente *table*.

Fin qui la differenza rispetto all'ambiente *table* consiste solo nel fatto che viene scritto nel titolino la parola 'Figura' invece della parola 'Tabella'. In realtà il contatore delle figure è diverso da quello delle tabelle, così come la coda delle figure, o meglio, la gestione della coda delle figure è diversa e distinta da quella delle tabelle.

Le figure possono venire separate dal testo che precedono o che seguono da appositi filetti `\topfigrule`, `\botfigrule` o `\dblfigrule` descritti meglio nel capitolo 29. Normalmente questi filetti sono assenti, ma possono essere resi visibili e contribuiscono non poco alla separazione del testo dalla didascalia, specialmente per le figure collocate in testa alla pagina.

9.2.1 Controllo dei grandi oggetti flottanti

Ci si ricordi solo che oggetti troppo grandi, siano essi figure o tabelle, possono bloccare il deflusso degli oggetti della stessa specie dalla rispettiva coda e questo potrebbe portare non solo allo 'scarico' di tutti questi oggetti alla fine del capitolo o del documento, ma potrebbe portare anche alla saturazione delle code e quindi ad una situazione di stallo nella esecuzione della compilazione del documento.

Un modo di evitare questo stallo è quello di usare il pacchetto esterno *afterpage* al quale si è già fatto cenno nel capitolo precedente; esso mette a disposizione del compositore il comando `\afterpage` con la seguente sintassi:

```
\afterpage{⟨azione⟩}
```

dove ⟨*azione*⟩ rappresenta il comando da eseguire alla fine della pagina corrente, quando, cioè, la pagina corrente viene accodata al file di uscita. In quel momento, subito dopo l'accodamento, LaTeX può eseguire l'⟨*azione*⟩; nella fattispecie l'⟨*azione*⟩ prescritta potrebbe essere `\clearpage` che, oltre a terminare la pagina corrente (già terminata) serve anche per scaricare le code. Se si specifica spesso

```
\afterpage{\clearpage}
```

si ottiene il risultato di non accumulare troppo materiale nelle code delle figure e delle tabelle; magari durante la revisione delle bozze si può decidere se e come modificare figure o tabelle per rendere il deflusso dalle code un poco più ordinato in modo che non sia necessario ricorrere ad `\afterpage`.

9.2.2 Modifica degli ambienti flottanti

La rete e gli archivi CTAN contengono una moltitudine di pacchetti di estensione che consentono di gestire o di creare nuovi oggetti flottanti.

Il pacchetto principale per questo scopo è *float* la cui documentazione si trova in `.../doc/latex/float/float.pdf`. Esso tra le altre cose consente di usare un nuovo codice di posizionamento, `H`, che impone a LaTeX di collocare 'qui' l'oggetto 'senza fare tante storie'. Sembra che la possibilità di poter controllare

la posizione precisa degli oggetti flottanti sia un desiderio fortissimo degli utenti di LaTeX; in realtà, salvo pochi rari casi, è meglio lasciare fare a LaTeX il suo lavoro, anche se può sembrare un po' rigido nelle scelte che esso compie nel collocare gli oggetti flottanti; per capirne meglio il meccanismo conviene riferirsi al capitolo 29.10 dove è descritto in dettaglio il meccanismo seguito da LaTeX per trovare la posizione corretta degli oggetti flottanti. Agendo sui parametri che controllano queste posizioni si può fare molto meglio che non specificando il parametro `H`.

Ma quel che più conta è che il pacchetto *float* consente di modificare le caratteristiche stilistiche degli ambienti flottanti e di definirne di nuovi; per esempio per esporre brani di programmazione, per descrivere degli algoritmi, insomma per collocare fuori testo qualunque cosa che possa essere reperita con un nome, un numero e una didascalia. Ognuno di questi ambienti flottanti può dare luogo a nuove liste/indici di oggetti flottanti alla stessa maniera con la quale si stampano gli elenchi delle figure e delle tabelle rispettivamente con i comandi `\listoffigures` e `\listoftables`.

Una ulteriore utile estensione è costituita dal pacchetto *floatrow*, la cui documentazione si trova in `.../doc/latex/floatrow/floatrow.pdf`; questo pacchetto consente di fare tutte le cose che si possono fare con il pacchetto *float* e molte altre ancora, al punto che se si usa *floatrow* non si deve caricare né *float* né *rotfloat* (quest'ultimo dedicato alla composizione di oggetti flottanti rotati di 90°) perché il codice di questi pacchetti è stato già incorporato in *floatrow*. Quest'ultimo consente anche di collocare diversi (piccoli) oggetti flottanti, ognuno con la sua didascalia, in fila uno accanto all'altro (da cui il nome del pacchetto) e consente persino di inserire un (piccolo) oggetto flottante insieme al testo che lo affianca o che lo circonda.

Va detto che è giusto poter disporre delle possibilità descritte sopra, ma questi strumenti devono essere usati con molta cautela, perché il risultato, tecnicamente perfetto, potrebbe avere una estetica molto discutibile. Per esempio, se si vogliono inserire due figure, ognuna con la sua didascalia, in un unico ambiente flottante, è bene che esse abbiano le stesse dimensioni e che le didascalie abbiano lo stesso numero di righe; se così non fosse, l'aspetto complessivo delle due figure affiancate sarebbe esteticamente assai dubbio.

9.3 L'ambiente `picture`

LaTeX dispone di un meccanismo per produrre disegni al tratto, semplici schemi a blocchi, semplici diagrammi e simili disegnini mediante l'ambiente *picture*.

L'ambiente *picture* sta a *figure* come *tabular* sta a *table*; forse questo paragone serve a comprendere meglio la differenza fra figura e immagine.

L'ambiente *originale*, nato insieme al vecchio LaTeX 2.09, era molto limitato nella scelta delle linee che potevano essere tracciate, perché venivano usati dei segmentini tratti da una polizza di font specialmente predisposti per questo

scopo; per quanto potessero essere ben disegnati, questi segmentini erano pur sempre in numero limitato.

Dal 2003 esiste l'estensione standard *pict2e* di LaTeX che non è ancora direttamente incorporata nel nucleo di LaTeX, ma che dovrebbe diventarlo in non molto tempo; questa estensione è progettata per eliminare tutte le limitazioni del vecchio ambiente e per aggiungere prestazioni nuove ed è per questo che nel seguito ci riferiremo solo a quanto si può fare con questo pacchetto. Si invocherà questa estensione mettendo nel preambolo il codice seguente:

```
\usepackage[⟨opzioni⟩]{pict2e}[2009/06/01]
```

dove ⟨*opzioni*⟩ può essere una o più fra le opzioni disponibili, la maggior parte delle quali serve per definire il tipo di driver che verrà usato per tradurre il risultato in una forma comprensibile agli umani. Per lo più servono solo in casi particolarissimi e rari, ma ci sono altre tre opzioni interessanti:

original serve a impostare *pict2e* in modo che sia perfettamente equivalente al vecchio LaTeX; evidentemente serve solo per compatibilità con il passato.

ltxarrows serve per specificare a *pict2e* che le punte delle frecce devono essere disegnate come vengono disegnate da LaTeX in modo di compatibilità con il passato; siccome quelle frecce con punta triangolare leggermente concave sono molto gradevoli, il compositore può decidere di usare tali punte di freccia, anche se non introducono nessun elemento di novità nello stile del disegno.

pstarrows serve invece a disegnare le punte delle frecce come un aereo stealth, cioè come un aereo con le ali a V; questa è la maniera di disegnare le frecce usata anche dal pacchetto PSTricks, eccellente pacchetto di estensione che si affida completamente al driver dvips per produrre un file di tipo PostScript.

Se non si specificano opzioni, il pacchetto cerca se esiste il file di configurazione `pict2e.cfg` e sceglie le opzioni che quel file riesce a definire in base al programma eseguito; inoltre sceglie di default le frecce tradizionali di LaTeX.

A parte questi preliminari, l'ambiente *picture* definisce uno spazio grafico in cui l'unità di misura è definita dal parametro `\unitlength`, che di default vale 1 pt. Specificando un valore diverso è facilissimo scalare il disegno in modo da ingrandirlo o da rimpicciolirlo. Come trucco personale io preferisco definire per ogni ambiente *figure*, prima di aprire l'ambiente *picture*, una unità di misura dipendente dal font in uso, per esempio 1 ex, così che cambiando il corpo del font anche il disegno viene scalato proporzionalmente. In altre circostanze scelgo come unità di misura un centesimo della giustezza; in questo modo sono sicuro che, se non supero il valore di 100 unità in orizzontale, il disegno certamente non esce fuori dei margini. La sintassi da usare per impostare l'unità di misura è:

```
\setlength{\unitlength}{⟨lunghezza⟩}
```

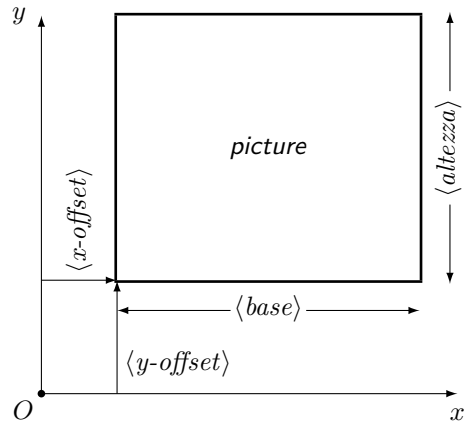


Figura 9.1: Relazione fra le dimensioni della figura e l'origine degli assi; i due offset possono anche essere negativi, così che l'origine degli assi cada dentro la figura.

Dopo questa prima operazione l'ambiente *picture* con le sue dimensioni apparenti e il suo offset, nonché ogni distanza o lunghezza a cui si faccia riferimento all'interno di questo ambiente, è indicata mediante un numero, senza specificare le unità di misura, perché queste, è sottinteso, sono identificate da `\unitlength`.

La sintassi di apertura dell'ambiente *picture* è la seguente:

```
\begin{picture}(⟨base⟩,⟨altezza⟩)(⟨x-offset⟩,⟨y-offset⟩)
...
\end{picture}
```

Si notino le parentesi tonde per specificare i multipli dell'unità di misura che si intendono usare; la prima coppia di parentesi tonde contiene le dimensioni della base e dell'altezza 'apparenti' del disegno; *apparenti* significa che il disegno verrà inserito dentro un rettangolo di quelle dimensioni, ma alcune parti del disegno potrebbero fuoriuscire da quel rettangolo, al punto che sarebbe possibile (e di fatto lo è) dichiarare entrambe le dimensioni pari a zero. La seconda coppia di parentesi è facoltativa; se c'è, essa indica la posizione dello spigolo inferiore sinistro del suddetto rettangolo rispetto all'origine degli assi x e y, come si vede nella figura 9.1; questi sono gli assi rispetto ai quali vengono specificate le coordinate degli oggetti da collocare nel disegno.

Ogni oggetto da collocare nel disegno viene collocato mediante il comando `\put`; se l'oggetto deve essere collocato un certo numero di volte in posizioni regolarmente spaziate si può usare il comando `\multiput`; le sintassi sono le seguenti:

```
\put(⟨x⟩,⟨y⟩){⟨oggetto⟩}
\multiput(⟨x-ini⟩,⟨y-ini⟩)(⟨x-step⟩,⟨y-step⟩){⟨numero⟩}{⟨oggetto⟩}
```

Le coordinate ⟨*x-ini*⟩ e ⟨*y-ini*⟩ sono le coordinate del primo punto; le coordinate ⟨*x-step*⟩ e ⟨*y-step*⟩ sono gli incrementi da dare ad x e ad y ogni volta che si deve inserire una nuova istanza dell'oggetto; il ⟨*numero*⟩ indica il numero totale di istanze dell'oggetto; infine ⟨*oggetto*⟩ è l'oggetto grafico da collocare nel disegno.

Gli oggetti collocabili sono linee, vettori (frecce), cerchi, dischi, testi semplici o riquadrati con linee continue o tratteggiate; i testi possono essere collocati intelligentemente rispetto al rettangolo ideale che li contiene. Possono poi essere collocati degli 'ovali', cioè dei rettangoli con gli spigoli arrotondati, dentro ai quali può essere collocato del testo; possono essere fatti semplici disegni tracciando delle curve mediante l'uso di curve di Bézier di secondo o di terzo grado. La sintassi che descrive ognuno di questi oggetti è la seguente:

```
\line(⟨x-pend⟩,⟨y-pend⟩){⟨lunghezza⟩}
\vector(⟨x-pend⟩,⟨y-pend⟩){⟨lunghezza⟩}
\circle{⟨diametro⟩}
\circle*{⟨diametro⟩}
\makebox(⟨base⟩,⟨altezza⟩)[⟨posizione⟩]{⟨testo⟩}
\framebox(⟨base⟩,⟨altezza⟩)[⟨posizione⟩]{⟨testo⟩}
\dashbox{⟨lung-trattino⟩}(⟨base⟩,⟨altezza⟩)[⟨posizione⟩]{⟨testo⟩}
\oval[⟨raggio⟩](⟨base⟩,⟨altezza⟩)[⟨parte⟩]
```

I nomi dei parametri dovrebbero essere abbastanza chiari, ma vanno specificate alcune cose particolari.

La lunghezza dei segmenti e dei vettori è la componente orizzontale, cioè la proiezione sull'asse x del segmento o del vettore; se questo è completamente verticale allora e solo allora la lunghezza coincide con quella del segmento o del vettore.

I coefficienti di pendenza dei segmenti e dei vettori rappresentano rispettivamente le proiezioni lungo gli assi x e y di un tratto di segmento o di vettore; in pratica sono i coseni direttori del segmento o del vettore, o quantità ad esse proporzionali (con lo stesso fattore di proporzionalità; Per un segmento o un vettore che abbia una inclinazione di 60° rispetto all'orizzontale si possono specificare rispettivamente i valori 0.5 e 0.866; si otterrebbe lo stesso risultato se si specificasse 500 e 866; l'importante è che nessun coefficiente di pendenza superi in valore assoluto il valore 16383.99999, a causa del modo particolare che usa l'interprete `pdftex` per eseguire i suoi calcoli interni. Per i segmenti, che non hanno una punta di freccia che ne indichi la direzione, questa va intesa come la direzione del segmento dal suo estremo collocato con `\put` all'altro estremo; in altre parole il comando `\put` colloca alla sua coordinata il primo punto del segmento, da cui parte il segmento con la pendenza specificata.

Il comando `\circle` specifica una circonferenza completa di cui si specifica il ⟨*diametro*⟩; il suo punto di riferimento messo in posizione con `\put`, è il suo centro. Il comando `\circle*` specifica invece un disco completo, non solo il contorno, quindi disegna un cerchio 'nero' con il ⟨*diametro*⟩ specificato.

I comandi `\makebox`, `\framebox` e `\dashbox` definiscono tre scatole, la prima senza che ne venga disegnato il contorno, la seconda con il contorno disegnato, la terza con il contorno tratteggiato con trattini lunghi ⟨*lung-trattino*⟩. Il comando `\put` mette in posizione il loro punto di riferimento, cioè lo spigolo inferiore sinistro. All'interno di questi rettangoli, o scatole, può essere posto del testo che viene collocato rispetto ai bordi ideali, disegnati o tratteggiati conformemente ai parametri di ⟨*posizione*⟩; questi sono le solite lettere `t`, `b`, `l`, `r` e `c`, con il rispettivo significato di 'top', 'bottom', 'left', 'right' e 'center'; per ogni oggetto si possono specificare al massimo due parametri di posizione[1], ricordando che 'center' è sempre il valore di default.

È particolarmente comodo il comando `\makebox` con dimensioni di ⟨*base*⟩ e ⟨*altezza*⟩ nulle, perché il ⟨*testo*⟩ che vi viene inserito, risulta collocato con precisione rispetto allo spigolo inferiore sinistro, ma senza spazio alcuno interposto; non ha importanza che le dimensioni apparenti della scatola siano nulle; è importante che il testo sia collocato con un riferimento preciso, indipendentemente dalla presenza di ascendenti o discendenti.

Per il comando `\oval` è possibile specificare il raggio minimo del quarto di cerchio che costituisce lo spigolo; le dimensioni di ⟨*base*⟩ e ⟨*altezza*⟩ di questo speciale rettangolo a spigoli arrotondati si riferiscono al rettangolo completo; ma la specifica facoltativa di ⟨*parte*⟩ permette di scegliere quali angoli mostrare o quale coppia di angoli adiacenti mostrare, così che specificando raggio, metà della base e metà dell'altezza uguali è possibile disegnare solo un quarto di circonferenza oppure metà circonferenza; al solito la ⟨*parte*⟩ è specificata con le lettere `l`, `t`, `b` e `r` (`c` non avrebbe senso), cosicché `t` permette di disegnare solo la metà di sopra, `tl` solo il quarto in alto a sinistra.

Nell'ambito dell'ambiente *picture* è possibile specificare due spessori predefiniti per le linee da tracciare, `\thinlines` e `\thicklines`; si può specificare un valore assoluto di spessore (espresso con le unità di misura) mediante:

```
\linethickness{⟨spessore assoluto⟩}
```

e questo comando modifica lo spessore sia delle linee rette e comunque inclinate, sia delle linee curve, siano esse archi di circonferenza oppure curve di Bézier quadratiche o cubiche.

Infine i comandi `\qbezier` e `\cbezier` consentono di mettere in posizione delle curve di secondo o di terzo grado rispettivamente, dette *spline* di Bézier, senza bisogno di ricorrere ai comandi `\put` o `\multiput`. La loro sintassi è la seguente:

```
\qbezier(⟨x1⟩,⟨y1⟩)(⟨x2⟩,⟨y2⟩)(⟨x3⟩,⟨y3⟩)
\cbezier(⟨x1⟩,⟨y1⟩)(⟨x2⟩,⟨y2⟩)(⟨x3⟩,⟨y3⟩)(⟨x4⟩,⟨y4⟩)
```

[1] Ovviamente non contrastanti. È ovvio che non si possono specificare simultaneamente `t` e `b`, perché lo stesso oggetto non può essere collocato allo stesso tempo in testa e al piede dentro la scatola specificata.

Per la curva di secondo grado, descritta da `\qbezier`, (x_1, y_1) rappresentano le coordinate del punto da cui la curva parte in direzione del punto avente le coordinate (x_2, y_2); invece (x_3, y_3) rappresentano le coordinate del punto di arrivo con la direzione che proviene da (x_2, y_2).

Per la curva di terzo grado, descritta da `\cbezier`, questa parte da (x_1, y_1) in direzione (x_2, y_2) e arriva al punto (x_4, y_4) con la direzione proveniente da (x_3, y_3).

Scegliendo accuratamente i punti iniziali e finali, nonché le direzioni delle tangenti si possono facilmente disegnare semplici diagrammi come mostrato nelle figure 9.2 e 9.3. I punti guida delle curve di Bézier non compaiono perché cadono fuori dal rettangolo della figura.

Va notato che nel 2009 il pacchetto *pict2e* è stato fortemente arricchito di nuove funzionalità che non sono delle semplici aggiustatine agli stessi comandi della versione tradizionale di LaTeX, ma sono delle estensioni decisamente importanti.

Innanzi tutto è stata definita una nuova routine di divisione fra numeri fratti che permette di dividere numeri di qualsiasi grandezza (purché inferiori in modulo a 16 384, sia dividendo, sia divisore, sia quoziente); per cui la restrizione sul fatto che inizialmente (nella versione di *pict2e* del 2004) i coefficienti di pendenza di segmenti e vettori dovessero essere rappresentati da numeri interi inferiori in modulo a 1000, è stata rimossa; ora si possono usare tranquillamente numeri fratti e, se per esempio si vuole una pendenza di 37°, il cui coseno vale 0,79864 e il cui seno vale 0,60182, si possono usare direttamente questi due numeri senza dover ricorrere all'artificio di moltiplicarli per 1000 e poi usare la sola parte intera arrotondata; non cambia molto da un punto di vista pratico, ma evita di dover fare calcoli e arrotondamenti.

Inoltre sono stati aggiunti altri comandi, alcuni dei quali tracciano il loro disegno quando siano messi come argomento di `\put`, e altri che sono assoluti, e cioè, come per le curve di Bézier, non hanno bisogno dell'intermediario del comando `\put`.

Comandi da usare con `\put`:

`\arc[`⟨⟨*angolo1*⟩,⟨*angolo2*⟩⟩`]{`⟨*raggio*⟩`}`
`\arc*[`⟨⟨*angolo1*⟩,⟨*angolo2*⟩⟩`]{`⟨*raggio*⟩`}`

Comandi che non necessitano di `\put`:

`\Line(`⟨x_1⟩`,`⟨y_1⟩`)(`⟨x_2⟩`,`⟨y_2⟩`)`
`\polyline(`⟨x_1⟩`,`⟨y_1⟩`)...(`⟨x_n⟩`,`⟨y_n⟩`)`
`\polygon(`⟨x_1⟩`,`⟨y_1⟩`)...(`⟨x_n⟩`,`⟨y_n⟩`)`
`\polygon*(`⟨x_1⟩`,`⟨y_1⟩`)...(`⟨x_n⟩`,`⟨y_n⟩`)`

I comandi `\arc` con o senza asterisco generano un arco con il centro nella coordinata specificata dalle coordinate del comando `\put`, di raggio pari a ⟨*raggio*⟩, che inizia nell'angolo ⟨*angolo1*⟩ e termina nell'angolo ⟨*angolo2*⟩; gli angoli sono specificati in gradi e con il segno positivo si indica la rotazione in

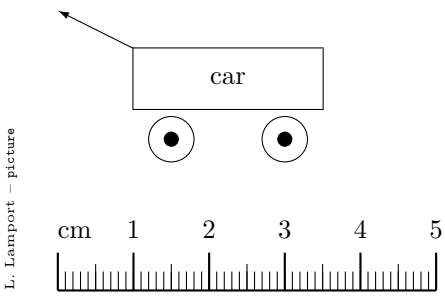


Figura 9.2: Il semplice disegno usato da Leslie Lamport per descrivere le potenzialità dell'ambiente *picture*

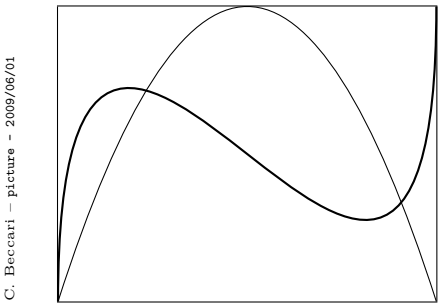


Figura 9.3: Alcune curve di Bézier di secondo e di terzo grado tracciate nell'ambiente *picture*

senso antiorario. La versione asteriscata riempie di colore (di default nero) il settore sotteso dall'angolo.

Il comando `\polyline` genera una spezzata che collega in successione gli n punti (x_1, y_1), (x_2, y_2), ..., (x_n, y_n); il numero di punti è arbitrario, perché il comando itera il proprio funzionamento fino ad esaurire la lista delle coordinate.

Il comando `\polygon` genera una spezzata chiusa nel senso che arrivato al punto n-esimo aggiunge ancora un segmento fino il primo punto della lista. Il comando `\polygon*` si comporta nello stesso modo ma riempie di colore (di default nero) l'interno del poligono.

Inoltre sono stati definiti nuovi comandi per specificare come devono essere formati gli estremi dei segmenti (tagliati dritti o arrotondati) e come debbano essere raccordati fra di loro i segmenti che formano le spezzate (se si prolungano fino ad incontrarsi i lati che delimitano i due segmenti da unire, oppure se questo prolungamento viene troncato ad una certa distanza dal punto di intersezione dei loro assi); per i dettagli si veda la documentazione. Si noti che con linee sottili questi dettagli sono virtualmente impercettibili, ma con linee relativamente spese la differenza c'è e si vede. Ricordiamoci che con *pict2e* si possono variare a piacere gli spessori di tutte le linee, quindi non è difficile ottenere segmenti di un certo spessore; semplicemente si cerchi di non dimenticare che una linea di uno spessore di circa mezzo millimetro ($\approx 1{,}5\,\mathrm{pt}$) è una linea nerissima; si veda la figura 9.4.

Usando i colori e i nuovi comandi di *pict2e* non è difficile disegnare il diagramma della figura 9.4.

Il codice per disegnare la figura 9.4 è il seguente:

```
\begin{figure}\unitlength=0.85cm\centering
\begin{picture}(11,11)(-1,-1)
\linethickness{0.35pt}
\multiput(0,0)(0.5,0){21}{\line(0,1){10}}
\multiput(0,0)(0,0.5){21}{\line(1,0){10}}
```

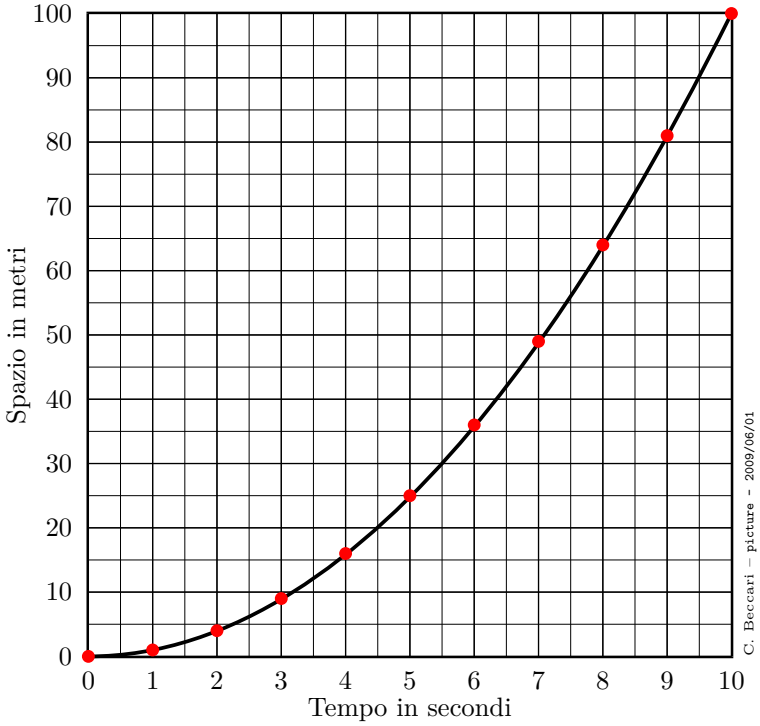


Figura 9.4: Moto uniformemente accelerato

```
\linethickness{0.5pt}
\multiput(0,0)(1,0){11}{\line(0,1){10}}
\multiput(0,0)(0,1){11}{\line(1,0){10}}
\linethickness{0.7pt}
\setcounter{cms}{0}
\put(0,0){\framebox(10,10){}}
\multiput(0,-0.5)(1,0){11}{%
   \makebox(0,0)[b]{\arabic{cms}}\stepcounter{cms}}
\put(10.2,0){\rotatebox{90}{%
   \tiny C. Beccari -- \texttt{picture - 2009/06/01}}}
\put(5,-1){\makebox(0,0)[b]{Tempo in secondi}}
\setcounter{cms}{0}
\multiput(-0.25,0)(0,1){11}{%
   \makebox(0,0)[r]{\arabic{cms}\addtocounter{cms}{10}}}
\put(-0.9,5){\rotatebox{90}{\makebox(0,0)[b]{Spazio in metri}}}
\linethickness{1.4pt}
\cbezier(0,0)(4.5,0)(8,6)(10,10)
\color{red}
\setcounter{cms}{-1}
```

```
\makeatletter
\multiput(0,0)(1,\strip@pt\dimexpr\value{cms}\p@/10\relax){11}%
    {\circle*{0.2}\addtocounter{cms}{2}}
\makeatother
\end{picture}
\caption{Moto uniformemente accelerato}\label{fig:motoaccelerato}
\end{figure}
```

Vi sono molti aspetti un po' particolari, come per esempio l'uso delle scatole `\makebox` con le dimensioni dichiarare nulle; questo è un espediente che si usa spesso e serve per poter collocare il contenuto della scatola in una posizione precisa rispetto ai suoi bordi (invisibili), mantenendo nello stesso tempo un completo controllo sul punto di riferimento dell'oggetto da mettere in posizione o anche da ruotare, come la legenda della scala delle ordinate.

Insomma, l'ambiente *picture* è semplice, ma con le estensioni introdotte nel pacchetto *pict2e* non è inferiore a molti altri software di disegno; ha il vantaggio che ogni informazione letterale o matematica che compare nel disegno è eseguita con gli stessi caratteri usati nel testo. Naturalmente non è l'unico mezzo per ottenere questo risultato, come d'altra parte si ricava da quanto esposto nel prossimo paragrafo.

9.4 Il pacchetto pgf

Il pacchetto di estensione *pgf* è ancora in evoluzione ma certamente le potenzialità che ha oggi verranno sicuramente mantenute ad ampliate in futuro. L'autore cerca di costruire una interfaccia utente che prescinda completamente dal particolare strumento informatico che trasforma il codice in qualcosa comprensibile dagli umani. Oggi si possono ottenere risultati ottimi con il pacchetto di estensione *PSTricks*, ma il suo codice è interpretabile solo dal driver dvips; per terminare con un file in formato PDF è necessario procedere ad una ulteriore conversione di formato[2].

L'acronimo PGF indica 'Portable Graphics Format' e vorrebbe poter un giorno essere altrettanto potente di *PSTricks* ma svincolato da questo o quel driver; nel momento in cui il sorgente `.tex` viene compilato, il motore di compilazione determina con quale programma questa compilazione viene eseguita e sceglie l'interfaccia giusta in relazione a quella situazione; se il file sorgente, come succede spesso, viene spedito per posta elettronica ad un conoscente che lavora con una piattaforma diversa e una distribuzione del sistema TEX diversa o è abituato ad

[2]Questa limitazione, che provoca una apparente incompatibilità fra il compilatore LaTeX e il compilatore pdfLaTeX, può essere aggirata attraverso il pacchetto *pst-pdf*, alla cui documentazione si rinvia il lettore; un altro pacchetto *auto-pst-pdf* offre una comoda interfaccia per l'uso di *pst-pdf* così che di fatto, sia pure con un lieve rallentamento della compilazione, anche il compilatore pdfLaTeX riesce a gestire file sorgente che fanno uso di *PSTricks*. Il lettore interessato può esplorare CTAN`/macros/latex/contrib/pst-pdf/` e CTAN`/macros/latex/contrib/auto-pst-pdf/`.

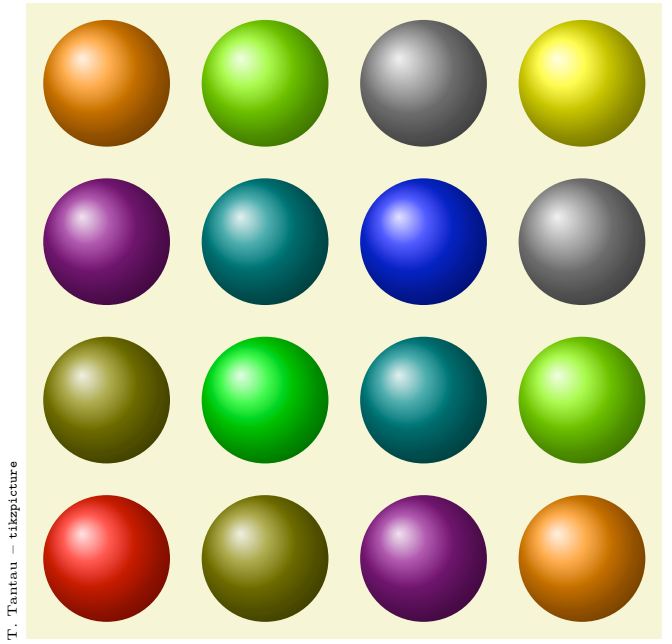


Figura 9.5: Un disegno a colori ottenuto con l'uso del pacchetto *pgf* e l'ambiente *tikzpicture*

usare un compilatore diverso, ebbene la grafica deve risultare del tutto identica a come riusciva all'autore originale.

Qui sarebbe fuori luogo scendere nei dettagli, e si rinvia il lettore alla documentazione del pacchetto *pgf*; sappia comunque il lettore che questo pacchetto mette a disposizione un ambiente di disegno che si chiama *tikzpicture*, all'interno del quale la sintassi dei comandi è molto diversa dalla sintassi da usare nell'ambiente *picture*, ma sotto certi aspetti essa è più intuitiva, in ogni caso più adatta alla moltitudine di oggetti, linee, colori, nodi, alberi, strutture, schemi di flusso, eccetera, che possono essere gestiti con quel pacchetto e quell'ambiente.

Qui, giusto per far venire l'acquolina in bocca con un blando aperitivo, si mostra solo la figura 9.5 che mostra una figura a colori, dove i colori delle sfere che giacciono sulla diagonale secondaria (quella che va da SW a NE) sono mescolati fra di loro nelle posizioni fuori da quella diagonale, mostrando sempre delle mezze sfere con gradienti di luce obliqua a simmetria circolare.

L'ambiente *tikzpicture* è estensibile con diversi altri ulteriori pacchetti; per esempio, *circuitikz* che permette di disegnare circuiti elettronici scegliendo lo stile americano oppure quello europeo; la figura 9.6 mostra un esempio tratto dal manuale del pacchetto che si trova in `.../doc/latex/circuitikz`. Il manuale espone anche l'intera lista di componenti, veramente molto ricca, con cui si possono disegnare circuiti elettronici; essendo ogni componente disegnato mediante

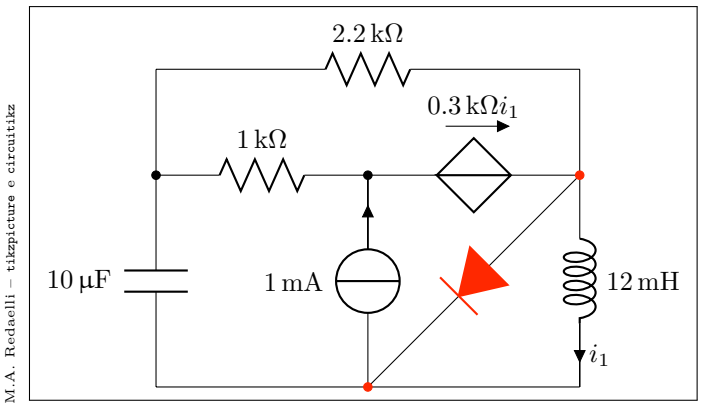


Figura 9.6: Circuito elettronico disegnato usando l'ambiente *tikzpicture* avendo caricato l'estensione *circuitikz*

una macro, non è difficile, casomai ce ne fosse bisogno, arricchire questa enorme collezione di ulteriori segni particolari.

L'ambiente *tikzpicture* e i comandi del pacchetto *tikz* possono venire usati per la definizione di diverse macro; in questo testo esso è stato usato per definire le macro che disegnano le icone che compaiono sui tasti di controllo della tastiera del portatile Mac Book Pro; esse possono venire esaminate nel file `MacroGuida.sty`.

Un altro semplice esempio suggerito da Antonio Cervone è il disegno per definire una paginetta con l'angolo superiore ripiegato "morbidamente"; questo disegno appare nell'Arte di scrivere con LaTeX, [49], e che tanto attrae anche i frequentatori del Forum del gruppo GuIr. Si è rielaborato il codice proposto da Antonio Cervone in modo da creare una macro che funziona solo con una versione moderna di pdflatex perché vi vengono usate delle funzionalità estese. Il codice rielaborato è il seguente:

```
\newlength\paginettaunit

\newcommand\paginetta[2][.5\textwidth]{{%
\countdef\Y=1000\countdef\YY=1001\relax
\setbox0\vbox{\hsize=#1\relax#2}\relax
\dimen0=#1\relax\dimen0=1.2\dimen0\relax
\paginettaunit=\dimexpr\dimen0/100\relax
\dimen0=\ht0
\Y=\numexpr\dimen0/\paginettaunit\relax
\YY=-\numexpr\Y+40\relax
\Y=-\numexpr\Y/2+27\relax
\begin{tikzpicture}[thick,x=\paginettaunit,y=\paginettaunit]
\draw[fill=blue!10]
  (0,0) [rounded corners=7.5] --
```

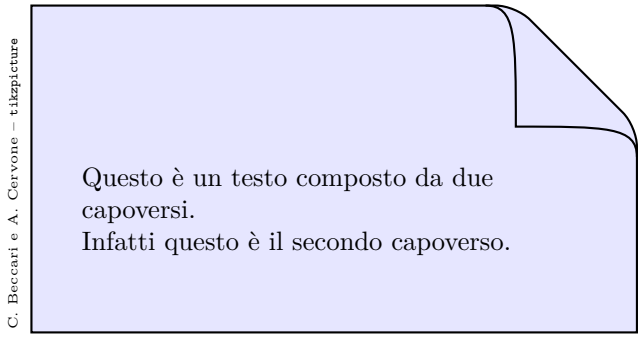


Figura 9.7: Una paginetta con l'angolo ripiegato

```
  (80,0) --
  (100,-20) [rounded corners=0] --
  (100,\YY) --
  (0,\YY) --
  cycle;
\draw
  (75,0) .. controls (80,0) and (80,-5) ..
  (80,-20) .. controls (95,-20) and (100,-20) ..
  (100,-25);
\node at (50,\Y) {\box0};
\end{tikzpicture}}}
```

e il risultato dell'uso del comando:

```
\paginetta[.55\textwidth]{\raggedright
Questo è un testo composto da due capoversi.

Infatti questo è il secondo capoverso.}
```

appare nella figura 9.7.

Nell'esempio della figura manca il rientro del capoverso; è voluto anche per mostrare quanto sia comodo quel rientro per capire subito che si sta leggendo un nuovo capoverso.

9.5 I vantaggi di usare programmi nativi del sistema TeX

I vantaggi di usare i programmi nativi del sistema TeX (*pgf* funziona anche con Plain TeX e con ConTeXt) consistono nel fatto che questi sistemi consentono di usare direttamente gli stessi font che vengono usati per il testo. Non si trascuri questa possibilità, perché l'uso di font diversi, talvolta incompatibili o non adatti

a quello che si sta scrivendo, è un difetto così grave che salta all'occhio anche della persona più inesperta.

I disegni, per esempio, prodotti con famosi programmi interattivi di matematica, possono essere salvati in formato PostScript; tuttavia è poi necessario correggere questi file per poter cambiare i font, specialmente quando alcune scritte che compaiono nei disegni sono costituite da formule e chiunque vede subito che non sono composte con la stessa maestria di quelle composte con LaTeX. Esistono dei pacchetti, per esempio *psfrag*, che riescono ad eseguire alcune di queste trasformazioni, ma resta sempre un lavoro maggiore da fare che richiede una attenzione maggiore, come succede sempre quando si eseguono delle correzioni.

Non si trascuri, quindi, la possibilità di disegnare semplici diagrammi ricorrendo anche al più elementare ambiente da disegno costituito da *picture*. Il vantaggio dell'uso dei font giusti supera qualunque fatica si debba fare per ottenere un risultato grafico accettabile. Se poi ci si prende il disturbo di imparare ad usare l'ambiente *tikzpicture* allora non si deve rinunciare proprio a niente.

9.6 METAPOST

Qui si vuol dare un ulteriore suggerimento al lettore interessato: egli può ricorrere al programma METAPOST per eseguire disegni al tratto sia bidimensionali sia tridimensionali.

Il programma METAPOST è descritto abbastanza dettagliatamente nel documento `.../doc/metapost/base/mpman.pdf`; è utile consultare anche il documento `.../doc/metapost/base/mpintro.pdf`; in ogni caso METAPOST prevede la conoscenza di METAFONT, quindi tutta l'operazione di documentazione ed apprendimento può sembrare piuttosto complessa. A questo proposito vale la pena di rompere il ghiaccio con `.../doc/english/metafp/metafp.pdf`, che è un testo introduttivo di Peter Wilson per iniziare ad usare sia METAFONT, sia METAPOST.

L'utilità di usare METAPOST per eseguire i disegni consiste nel fatto che si possono usare i costrutti più complessi che vengono elaborati da quel programma in modo che il file di uscita sia compatibile sia per il driver dvips sia per l'interprete pdftex (in pratica pdfLaTeX), cosicché il disegno prodotto può essere utilizzato sia per comporre un documento in formato PostScript sia in formato PDF senza bisogno delle successive trasformazioni da un formato all'altro. Si veda anche il paragrafo 10.3.

Un altro vantaggio dell'uso di METAPOST consiste nel fatto che tutte le indicazioni testuali e matematiche da inserire nel disegno vengono composte da LaTeX o pdfLaTeX stessi grazie ai comandi contenuti nel file sorgente METAPOST; l'uso dei font compatibili con quelli del testo composto è quindi assolutamente garantito.

A titolo di esempio si riporta il listato METAPOST del disegno che compare nella figura 9.8. I punti indicati con $\mathtt{z}_i$ sono alcune coordinate del piano che

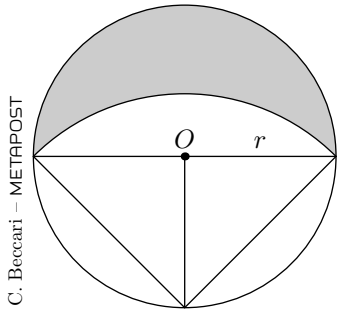


Figura 9.8: Un semplice esempio di figura geometrica composta con METAPOST

rappresentano i punti di appoggio per circonferenze e segmenti; le grandezze `pout` e `pin` sono dei *path*, cioè dei percorsi definiti tramite le precedenti coordinate e i comandi che li raccordano con curve di Bézier di terzo grado con concavità modeste (sequenze di tre puntini); i comandi `fill` e `draw` fanno esattamente quello che dice il loro nome. Interessanti sono i comandi per etichettare i punti: `dotlabel.top` disegna un punto nero nel punto `z0` e lo etichetta con quanto scritto fra parentesi. Invece `label.top` non disegna niente, ma mette l'etichetta contenuta fra parentesi a metà strada fra il punti `z0` e `z3`. Entrambi questi comandi per etichettare scrivono in linguaggio TeX quanto è contenuto fra `btex` ed `etex`; vi si leggono infatti le specificazioni per due oggetti matematici (da comporre con lo stile `\textstyle`); uno è il nome del centro del cerchio esterno O, e l'altro è l'indicazione del raggio r.

```
beginfig(1);
u:=1mm;
path pout,pin;
z1=(0,20u);
z2=(20u,40u);
z3=(40u,20u);
z4=(20u,20*sqrt(2)*u);
z0=.5[z1,z3];
pout= z1{up}...z2{right}...{down}z3;
pin= z3{z2-z3}...z4{left}...{z1-z2}z1;
fill pout&pin&cycle withcolor .8white;
z5=(20u,0);
draw pout...z5{left}...cycle;
draw pin;
draw z1--z3;
draw z0--z5;
draw z5--z1;
draw z5--z3;
dotlabel.top(btex$O$etex,z0);
```

```
label.top(btex$r$etex,.5[z0,z3]);
endfig;
end
```

Se il breve listato precedente è contenuto nel file `cerchi.mp` e, da linea di comando, si dà il comando[3]

```
mp cerchi
```

si ottiene il file `cerchi.001`, a cui si può cambiare nome in `cerchi.mps`, per poi importarlo nel documento da comporre con il comando `\includegraphics` di cui si parlerà nel prossimo capitolo[4].

9.7 Usi insoliti dell'ambiente *picture*

Le tabelle, come si è detto alla fine del capitolo 8, possono presentare dei problemi e si è indicato come risolverli, almeno in certi casi; la tabella 9.1 contiene sicuramente del materiale ordinato in celle giustapposte, ma le grandi parentesi graffe certamente disturbano gli allineamenti, perché si protraggono oltre i limiti delle celle che intendono raggruppare nelle varie colonne.

Se si compone la tabella nell'ambiente *picture* invece che nell'ambiente *tabular* bisogna rinunciare agli automatismi che quest'ultimo ambiente mette a disposizione, ma l'ambiente *picture* consente di collocare con precisione qualunque oggetto fornendo le coordinate del suo punto di riferimento, ma anche nascondendone le dimensioni reali.

Perciò con l'ambiente *picture* si possono parzialmente sovrapporre oggetti, pur di avere la pazienza di determinare le posizioni dei punti di collocamento con alcune prove. La tabella 9.1[5] è stata composta con il codice seguente, dove le coordinate da dare come argomento ai vari comandi `\put` sono state determinate sperimentalmente.

```
\newcommand*\rBrace[1]{%
                  \makebox(0,0)[r]{$\left.\rule{0pt}{#1}\right\}$}}
...
\setlength{\unitlength}{1ex}
\begin{picture}(85,60)
%
\put(0,40){$\left\{\parbox{25\unitlength}{\null
  Gravemente insufficiente\newline (saltare un appello) \[v<15\]
  Insufficiente\newline (può ripetere l'esame)\[ 15 \leq v<18\]
  Sufficiente \[ v\geq 18\]}\right.$}
```

[3] Il comando per lanciare METAPOST può essere `mp` oppure `mpost` a seconda dell'installazione del sistema TEX.

[4] Alternativamente si può usare il comando METAPOST `outputtemplate` descritto nella pagina 212.

[5] L'esempio è stato suggerito da Enrico Bellino, Università Cattolica del Sacro Cuore.

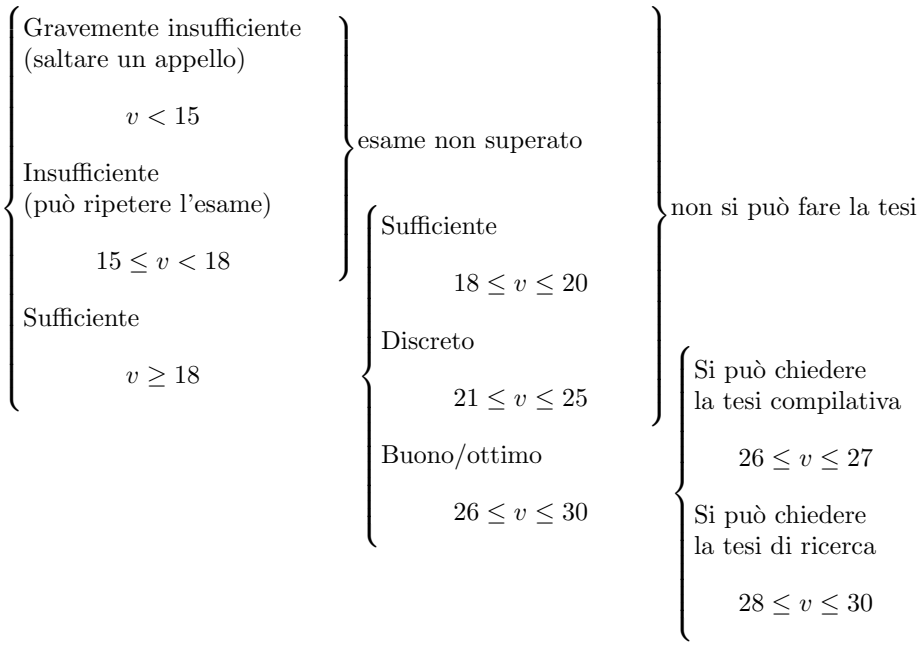


Tabella 9.1: Tabella insolita

```
%
\put(32,46){\rBrace{4.5\baselineskip}esame non superato}
%
\put(32,25){$\left\{\parbox{25\unitlength}{\null
  Sufficiente \[18\leq v\leq 20 \]
  Discreto \[ 21\leq v\leq 25\]
  Buono/ottimo \[26\leq v\leq 30\]}\right.$}
%
\put(60,40){\rBrace{7\baselineskip}non si può fare la tesi}
%
\put(60,14.5){$\left\{\parbox{20\unitlength}{\null
  Si può chiedere\newline la tesi compilativa \[26\leq v\leq 27\]
  Si può chiedere\newline la tesi di ricerca \[28\leq v\leq 30\]}
\right.$}
%
\end{picture}
```

La definizione che precede i tre puntini è nel preambolo; essa definisce il comando `\rBrace` che serve per comporre una parentesi graffa di semialtezza specificata, con il punto di riferimento al suo centro, ma racchiusa dentro una scatola di dimensioni nulle; in questo modo la parentesi graffa può essere collocata

ovunque senza che causi nessun intralcio al posizionamento degli altri oggetti: è come se non ci fosse.

Successivamente, prima di aprire l'ambiente *picture* (ma dentro all'ambiente *table* per rendere il tutto flottante e con il titolino della didascalia corretto) si assegna all'unità di misura `\unitlength` un valore dipendente dal font in uso: 1 ex.

Aperto l'ambiente *picture* si collocano i vari oggetti; si notino le scatole definite mediante `\parbox` che consentono di comporvi dentro del testo come se si trovasse dentro una colonna con la giustezza pari a quanto specificato nel primo argomento. Inoltre il primo elemento che vi si compone è la scatola nulla `\null`, la quale impegna una riga, del tutto bianca; serve per bilanciare lo spazio che si trova in fondo alla scatola dopo l'equazione centrata. Il resto di quanto si trova a formare il secondo argomento di `\parbox` è codice senza sorprese.

Si nota invece che le grandi parentesi graffe chiuse, costruite mediante il comando `\rBrace` e la semialtezza specificata, non occupando nessuno spazio, non alterano il collocamento del testo dell'unica riga che le accompagna.

La tabella 9.1 non è il massimo dell'eleganza, ma talvolta è necessario costruire tabelle di questo genere usate abitualmente in certe discipline scientifiche. In queste circostanze ricorrere all'ambiente *picture* sembra essere la soluzione più semplice.

9.8 Linee guida per la grafica

Per creare della grafica ben fatta bisogna avere talento e seguire alcune linee guida; con il permesso dell'autore, Till Tantau, qui si riproduce il capitolo 4 della prima parte del suo manuale [56] corredato di alcuni commenti messi in evidenza con il font di questo capoverso, in modo che si stacchino bene dal testo originale, sia pure tradotto con un certa libertà.

Si noti che nel seguito con la parola *grafica* si intende sia l'attività connessa con la produzione di disegni, sia il prodotto di questa attività, cioè uno o più disegni. Con *grafici* si intendono i disegni in senso stretto, generalmente disegni al tratto, quasi sempre privi di sfumature di colore, e che possono avere caratteristiche molto diverse.

I *diagrammi* sono disegni dell'andamento di una o più curve riferite ad assi cartesiani nel piano, meno frequentemente sono la proiezione sulla carta di diagrammi tridimensionali. Gli *schemi a blocchi* o i *diagrammi di flusso* sono disegni di scatole rettangolari o altre forme geometriche chiuse, spesso con sfondi colorati, collegati fra di loro da frecce che descrivono il fluire delle informazioni da una scatola all'altra. Gli *istogrammi* descrivono in due dimensioni, spesso con effetti tridimensionali, distribuzioni statistiche assai spesso non collegate ad una variabile indipendente misurabile. I *diagrammi a torta*, analogamente, descrivono delle distribuzioni statistiche mediante la lunghezza di archi di una circonferenza o l'area di settori di un cerchio. I *cartogrammi* rappresentano distribuzioni proiettate sul piano

di terreni o di aree geografiche distinte in modo che sia agevole risalire sia al loro perimetro, sia alla loro superficie, e includono le carte geografiche in senso stretto, le mappe e le planimetrie. I *pittogrammi* possono avere diversi aspetti e rappresentare le cose più diverse mediante aggregazioni di disegnini che rappresentano cose, animali o persone. I *disegni al tratto* più generali, non classificabili come descritto sopra, possono essere delle rappresentazioni schematiche o prospettiche della realtà, dove il disegno, privo di molta dell'informazione che potrebbe essere presente in una fotografia, permette di mettere in evidenza dei particolari, anche in trasparenza, che non sarebbero altrimenti evidenziabili.

In Italia la produzione di questi vari tipi di disegni è regolata dalle norme UNI [59]. Ciò non toglie che la scelta di un dato tipo di rappresentazione non possa essere sottoposta ad ulteriori regole di carattere estetico o di qualità della comunicazione.

9.8.1 Preliminari

Questo paragrafo non riguarda in senso stretto né PGF né Ti*k*Z, ma riguarda le linee guida generali e i principi che presiedono alla grafica per le presentazioni scientifiche, per gli articoli e per i libri.

Le linee guida in questo paragrafo arrivano da diverse fonti. Molte io le chiamerei semplicemente "buon senso", alcune riflettono la mia esperienza personale (sebbene, spero, non le mie preferenze personali), alcune arrivano da libri di tipografia e di disegno grafico, di cui purtroppo non ho composto la bibliografia. La fonte più importante è costituita dai libri di Edward Tufte. Sebbene io non sia d'accordo con ogni cosa scritta in questi libri, molte delle spiegazioni di Tufte sono così convincenti che ho deciso di riportarle fra queste linee guida [58].

La prima cosa che bisognerebbe domandarsi quando qualcuno propone una serie di linee guida è: "Dovrei davvero seguire queste linee guida?" Questa è una domanda importante, perché esistono dei buoni motivi per non seguire linee guida di carattere generale.

- La persona che ha scritto quelle linee guida potrebbe aver avuto degli obbiettivi diversi dai vostri. Per esempio una linea guida potrebbe affermare "usa il colore rosso per dare enfasi". Mentre questa linea guida potrebbe avere senso, per esempio, quando si predispone una presentazione da proiettare, il "colore" rosso ha un effetto *opposto* all' "enfasi" quando viene stampato con una stampante in bianco e nero.

  Le linee guida sono state quasi sempre scritte per affrontare una situazione specifica. Se non siete in quella situazione, seguire una linea guida può fare più male che bene.

- La regola base della tipografia dice che: "Ogni regola può essere infranta se siete *consapevoli* di ciò che state facendo". Questa regola vale anche per la grafica. Detto in modo equivalente, la regola base dice che: "I soli errori che si possono fare in tipografia sono le cose fatte per ignoranza."

> Quando conoscete una regola e quando ritenete che infrangere la regola produca l'effetto desiderato, allora infrangete la regola.

Perciò prima di seguire una linea guida o di decidere di non seguirla, fatevi le seguenti domande:

1. Questa linea guida riguarda davvero la mia situazione?
2. Se faccio l'opposto di quello che dice la linea guida, i vantaggi saranno superiori agli inconvenienti che la linea guida si propone di evitare?

9.8.2 Programmazione del tempo necessario per la creazione della grafica

Quando create un documento che contiene molta grafica, il tempo necessario per creare questa grafica diventa un fattore importante. Quanto tempo bisognerebbe stimare per la creazione della grafica?

Come regola generale supponete che la grafica vi richieda altrettanto tempo di quello necessario per produrre un testo della stessa lunghezza. Per esempio, quando io compongo un documento, impiego circa un'ora per ogni pagina della prima bozza. Poi per le revisioni impiego da due a quattro ore per ogni pagina. Perciò mi aspetto di avere bisogno di circa mezz'ora per creare la prima bozza di un grafico che occupi mezza pagina. Successivamente penso di avere bisogno ancora di una o due ore prima che il grafico sia a posto.

In molte pubblicazioni, anche in ottime riviste, gli autori e i revisori editoriali hanno evidentemente investito molto tempo sul testo, ma sembra che abbiano speso cinque minuti per creare tutta la grafica. I disegni sembra che siano stati aggiunti in un secondo tempo senza un motivo particolare; oppure sembrano una copia della schermata di qualunque cosa il software di statistica dell'autore gli mostrasse. Come verrà spiegato più avanti, la grafica prodotta da programmi come gnuplot con le impostazioni di default sono di qualità assai modesta.

Creare grafici che convogliano in modo corretto la loro informazione al lettore e che nello stesso tempo sono coerenti con il testo è un processo lento e difficile.

- Trattate i disegni come elementi di prima categoria. Essi meritano lo stesso tempo e la stessa energia che richiede il testo.
- Presumibilmente la creazione di grafici richiede *anche maggior tempo* di quello richiesto dalla composizione del testo, perché alla grafica si darà più importanza in quanto i disegni verranno osservati per primi.
- Pianificate di usare lo stesso tempo per la creazione e la revisione dei vostri grafici quanto ne dedichereste ad un testo della stessa estensione.
- Grafici difficili con un'alta densità di informazione vi possono richiedere anche più tempo.

- Grafici molto semplici vi possono richiedere meno tempo, ma verosimilmente voi non volete scrivere un documento con "grafici molto semplici"; per lo meno non lo volete come non vorreste scrivere "testi molto semplici" della stessa estensione.

9.8.3 Piano di lavoro per creare un grafico

Quando scrivete un articolo (scientifico), probabilmente seguite questo schema: avete dei dati o dei risultati da comunicare. La creazione dell'articolo tipicamente comincia con la predisposizione di una scaletta. Poi le varie sezioni vengono riempite con il testo per creare la prima bozza. Questa bozza subisce diverse revisioni e, spesso dopo revisioni sostanziali, finalmente diventa l'articolo finale. Tipicamente in un buon articolo per una rivista non c'è una sola frase che arrivi alla fine senza aver subito qualche modifica.

Creare la grafica segue lo stesso percorso:

- Decidete dapprima che cosa il grafico deve comunicare al lettore; prendete consapevolmente questa decisione, cioè domandatevi: "Questo grafico che cosa deve dire al lettore?"
- Preparate uno schema grossolano del grafico con la sua forma e i suoi punti essenziali; spesso va bene farlo con carta e matita. Spesso è utile abbozzare il disegno su carta millimetrata o almeno quadrettata.
- Inserite i successivi dettagli per produrre la prima bozza.
- Revisionate il grafico via via che rivedete il testo.

9.8.4 Collegamento fra testo e grafico

I grafici possono essere collocati in diverse posizioni rispetto al testo, nel senso che essi possono venire collocati in linea col testo, ovvero "in mezzo al testo", oppure essi possono essere usati come "figure" autonome. Siccome agli stampatori piace avere le pagine "piene", (per motivi che sono allo stesso tempo economici ed estetici), le figure autonome tradizionalmente possono venire collocate in posizioni relativamente distanti dal testo che vi fa riferimento. LaTeX sembra eseguire questo "allontanamento" per motivi tecnici, cioè per dividere automaticamente il testo in pagine, evitando pagine mozze e/o pagine con i capoversi malamente ed eccessivamente spaziati.

Quando un grafico è inserito in linea col testo, esso viene automaticamente collegato ad esso nel senso che le "etichette" del grafico vengono spiegate e descritte dal testo circostante. E il testo spiega chiaramente ciò a cui il grafico si riferisce e che cosa vi è mostrato.

All'opposto una figura autonoma spesso viene esaminata quando il testo che vi si riferisce o non è ancora stato letto, oppure è stato letto diverse pagine prima. Per questo motivo è bene seguire le seguenti linee guida quando si creano figure autonome o flottanti;

- Le figure flottanti devono avere una didascalia che permetta di renderle "autocomprensibili".

  Per esempio, supponete che un diagramma rappresenti gli stadi successivi dell'algoritmo *quicksort*. Allora la didascalia, come minimo dovrebbe informare il lettore che "La figura mostra gli stadi successivi dell'algoritmo *quicksort* spiegato nella pagina xyz." e non semplicemente "Algoritmo *quicksort*."

- Una buona didascalia aggiunge tanta informazione contestuale quanto è possibile. Per esempio, essa potrebbe specificare: "La figura mostra i diversi stadi dell'algoritmo *quicksort* spiegato nella pagina xyz. Nella prima linea l'elemento 5 è scelto come cerniera. Questo produce..." Anche se questa informazione può venire inserita nel testo, collocarla nella didascalia assicura che venga mantenuta la contestualità. Non abbiate paura di una didascalia di cinque righe. (Il vostro revisore editoriale vi odierà. Prendete in considerazione la possibilità di ricambiarlo.) Inoltre ricordatevi che la didascalia è composta di tre parti: il titolino corrente, il titolo e, opzionalmente, il testo; sfruttate la possibilità di usare questo testo; nello stesso tempo siate succinti per evitare che LaTeX incontri troppe difficoltà per far flottare la vostra figura in una posizione conveniente.

- Fate riferimento al grafico nel testo con una frase del tipo: "Per un esempio di *quicksort* "al lavoro" vedete la figura m.n nella pagina xyz."

- La maggior parte dei libri di tipografia raccomanda di non usare abbreviazioni del tipo "fig. m.n" ma di scrivere "figura m.n".

  L'argomentazione principale contro le abbreviazioni è che "un punto vale troppo per sprecarlo per una abbreviazione'. L'idea è che un punto può indurre il lettore a credere che la frase finisca subito dopo "fig" e che quindi gli sia necessario ritornare indietro coscientemente per rendersi conto che la frase non è affatto finita.

  L'argomentazione a favore delle abbreviazioni è che si risparmia spazio.

  Personalmente nessuna delle due argomentazioni mi convince. Da un lato non ho nessuna prova convincente che le abbreviazioni rallentino la lettura. Dall'altro anche se si abbreviano tutte le parole "figura" in "fig.", è veramente difficile che si risparmi anche una sola riga di testo.

  In alcuni libri di tipografia italiana si trova la raccomandazione di non usare abbreviazioni nel testo principale, ma se ne consente l'uso solo negli incisi, per esempio in quelli fra parentesi.

  Io evito le abbreviazioni.

9.8.5 Coerenza fra testo e grafica

Probabilmente "l'errore" che la maggior parte della gente fa quando crea dei grafici (ricordate che un "errore" nel disegno è sempre semplicemente "ignoranza") è quello di avere un disadattamento fra l'aspetto del grafico e quello del testo.

È abbastanza comune che gli autori usino diversi differenti programmi per creare i loro grafici per un articolo. Un autore potrebbe produrre alcuni diagrammi usando `gnuplot`, alcuni altri usando `xfig` e potrebbe includere alcuni grafici in formato `.eps` ottenuti da qualche collaboratore che ha usato programmi che gli sono ignoti. Tutti questi grafici, verosimilmente, useranno diversi spessori delle linee, font differenti, e avranno dimensioni diverse. Inoltre gli autori spesso usano opzioni come `height=5cm` quando includono dei grafici al fine di scalarli in modo accettabile.

Se si usasse il medesimo approccio per comporre il testo, ogni paragrafo verrebbe composto con un font diverso e di corpo diverso. In un paragrafo i teoremi verrebbero sottolineati, in un altro verrebbero composti in maiuscoletto, in un altro evidenziati in rosso. Inoltre i margini sarebbero diversi in ogni pagina.[6]

Né i lettori né i revisori editoriali tollererebbero un testo scritto in questo modo, ma con la grafica spesso sono costretti a farlo.

Per mantenere una consistenza fra testo e grafica seguite le linee guida seguenti:

- Non cambiate scala ai grafici; in altri termini non riproduceteli a dimensioni diverse da quelle con cui i diagrammi sono stati disegnati.

  Questo significa che quando usate programmi esterni per creare grafici, li dovete creare fin dall'inizio "delle dimensioni giuste".

- Usate gli stessi font per i grafici e per il testo.

- Usate lo stesso spessore delle linee per i grafici e per il testo.

  Lo "spessore delle linee" per il testo normale è in buona sostanza lo spessore delle aste di lettere come la T. Per i font del sistema TeX questo significa 0,4 pt. Tuttavia ci sono alcune riviste che non accettano grafici che contengano linee di spessore inferiore a 0,5 pt.

- Se usate i colori, usateli coerentemente seguendo un "codice dei colori". Per esempio, se usate il rosso nel testo per richiamare l'attenzione del lettore, usate lo stesso colore rosso per richiamare l'attenzione su parti importanti del disegno. Se il colore blu è usato per evidenziare le parti strutturali del testo, come per esempio i titoli dei paragrafi, usate lo stesso colore blu per evidenziare le parti strutturali dei vostri grafici.

  Tuttavia nella grafica si possono usare colori corrispondenti ad una codifica logica e "intrinseca". Per esempio, indipendentemente dal codice dei colori

[6] *Bisogna convenire che non è raro vedere testi composti in modo così sciatto, quando sono stati usati i soliti word processor; Con LaTeX è difficile che ciò accada, non impossibile, ma bisogna mettercela tutta...*

che voi possiate usare, i lettori generalmente associano il colore verde a concetti del tipo "positivo, va bene, OK", mentre associano il colore rosso a concetti come "attenzione, pericolo, azione".

Mantenere la coerenza quando si usano diversi programmi di disegno è quasi impossibile. Per questo motivo dovreste prendere in considerazione di usare sempre lo stesso programma.

9.8.6 Legende nei grafici

Praticamente tutti i disegni contengono delle etichette per individuare parti del grafico o legende testuali che spiegano parti del grafico. Quando usate queste legende, rimanete aderenti alle linee guida seguenti:

- Siate coerenti nel predisporre queste legende. Lo potete fare in due modi: primo, siate coerenti con il testo principale usando gli stessi font. Secondo, mantenete la coerenza fra una legenda e l'altra, cioè, se componete alcune legende in un certo modo, componente tutte le legende nello stesso modo.

- Oltre ad usare gli stessi font nel testo principale e nel grafico, dovreste anche usare le stesse notazioni. Per esempio, se scrivete 1/2 nel testo principale, usate la stessa scrittura nelle legende, non scrivete 0,5. Un π è un π e non 3,141. Ancora, $e^{-i\pi}$ è e resta $e^{-i\pi}$, non "-1" e meno che mai "-1".

- Le legende devono essere leggibili[7]. Esse pertanto devono essere abbastanza grandi e non devono essere disturbate dalla presenza di linee o di altro testo a loro in parte sovrapposto. Questo, ovviamente, vale anche per le linee e per il testo "dietro" alle legende.

- Le legende devono stare "nel posto giusto". Quando c'è abbastanza spazio, le legende dovrebbero essere collocate vicino all'oggetto a cui si riferiscono. Solo se è veramente necessario si può usare una leggera linea che connetta la legenda con l'oggetto. Evitate di usare legende che si riferiscono a spiegazioni contenute in legende esterne, altrimenti il lettore sarebbe costretto a saltare avanti e indietro fra la spiegazione e l'oggetto che viene descritto.

- Prendete in considerazione la possibilità di ridurre l'evidenza di alcune etichette meno importanti affinché il lettore possa focalizzare meglio la sua attenzione sul grafico.

9.8.7 Diagrammi di vario genere

Specialmente negli articoli scientifici, i grafici più frequenti sono i *diagrammi*. Possono essere composti in molte differenti maniere, che vanno dai semplici

[7] Legenda è una parola latina neutra plurale e significa "le cose che devono essere lette".

Valutazione	*Partecipanti (su 50) che diedero questa valutazione*	*Percentuale*
ottimo	3	6%
buono	9	18%
passabile	10	20%
scadente	8	16%
pessimo	0	0%
nessuna	20	40%

Tabella 9.2: Esempio di valutazione di un seminario

diagrammi con una sola curva, ai diagrammi parametrici, a quelli tridimensionali, ai diagrammi a torta, e molti altri.

È purtroppo ben noto che è difficile fare dei bei diagrammi. In parte bisogna incolpare le impostazioni di default di programmi come gnuplot o Excel, che consentono di disegnare facilmente brutti diagrammi.

La prima cosa che dovreste domandarvi quando volete creare un diagramma è la seguente:

- Ci sono abbastanza dati da riportare sul diagramma?

Se la risposta è "Per la verità, no", componente una tabella invece di un diagramma.

Una tipica situazione nella quale un diagramma non è necessario è quando si riportano pochi dati su un istogramma a barre. Ecco un esempio reale: Alla fine di un seminario l'oratore chiese ai presenti una valutazione. Dei 50 partecipanti, 30 restituirono il questionario di valutazione. La valutazione risultò essere la seguente: tre partecipanti dissero che il seminario era stato ottimo, nove lo considerarono buono, dieci lo considerarono passabile, otto scadente, mentre nessuno lo considerò pessimo.

Un modo semplice per raccogliere questa informazione è rappresentato dalla tabella 9.2.

Quello che, invece, l'oratore fece, fu di presentare il risultato sotto forma di un istogramma a barre, come mostrato nella figura 9.9.

Si osservi che la tabella 9.2 e la figura 9.9 occupano pressappoco lo stesso spazio. Se il vostro primo pensiero è che "il grafico è più gradevole della tabella", provate a rispondere alle seguenti domande basandovi sull'informazione contenuta nel grafico o nella tabella.

1. Quanti partecipanti c'erano?

2. Quanti partecipanti restituirono il questionario?

3. In che percentuale i partecipanti restituirono il questionario?

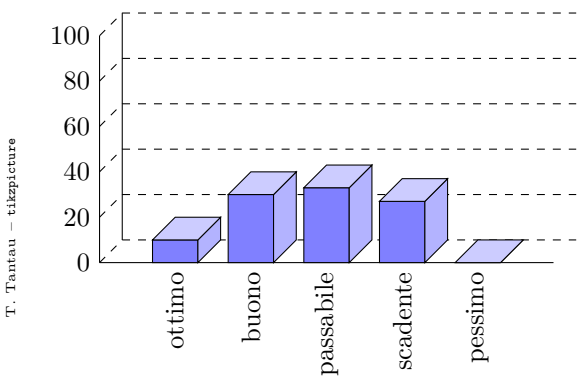


Figura 9.9: Esempio di istogramma per presentare il risultato della valutazione di un seminario

4. Quanti partecipanti risposero “ottimo”?

5. Fra tutti i partecipanti, quale percentuale rispose “ottimo”?

6. Furono più di un quarto coloro che risposero “scadente” o “pessimo”?

7. Fra coloro che restituirono il questionario, quale fu la percentuale di coloro che valutarono il seminario “ottimo”?

Purtroppo l’istogramma non permette di rispondere a *nemmeno una di queste domande*. La tabella risponde direttamente a tutte le domande, tranne l’ultima, per la quale, inoltre, dà gli elementi per eseguire il calcolo. In sostanza la densità di informazione dell’istogramma è quasi nulla; la tabella ha una densità di informazione molto maggiore, anche se contiene una grossa parte di spazio bianco e pochi numeri.

Ecco che cosa è andato storto con l’istogramma:

- L’intero grafico è dominato dalla presenza fastidiosa delle linee di sfondo.
- Non è chiaro il significato dei numeri a sinistra; probabilmente si tratta di percentuali, ma potrebbe anche essere il numero assoluto dei partecipanti.
- Le legende in basso sono ruotate, così che diventano più difficili da leggere.

  (Nella presentazione reale che io ho visto, il testo era composto con caratteri a matrici di punti campionati su una griglia di non più di 10 pixel per 6 pixel per ogni lettera, con accostamenti errati al punto da risultare quasi illeggibile.)
- La terza dimensione aggiunge al diagramma una certa complessità ma non aggiunge nessuna informazione.

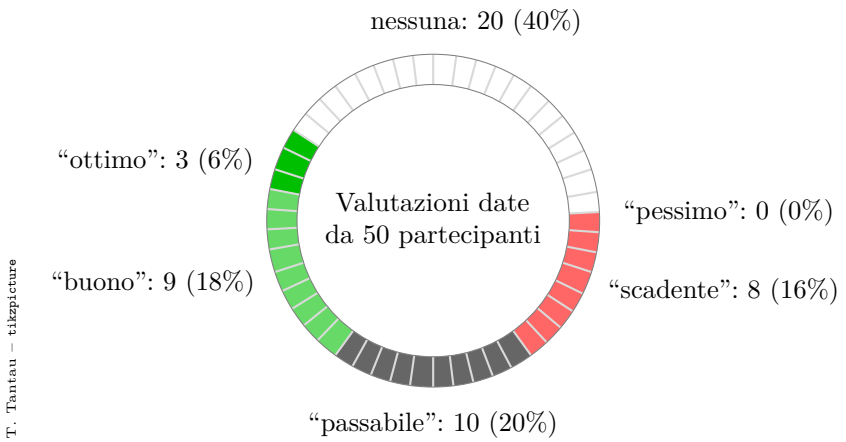


Figura 9.10: Altro esempio di diagramma per presentare il risultato della valutazione di un seminario

- L'impostazione tridimensionale rende molto più difficile da leggere l'altezza delle barre. Esaminate la barra “scadente”. La sua altezza è maggiore o minore di 20? Mentre la parte frontale della barra è sotto al livello 20, il retro della barra, che è quello che conta, è sopra a tale livello.
- È impossibile dire quali numeri siano rappresentati dalle barre. Perciò le barre nascondono senza necessità proprio l'informazione che dovrebbero dare.
- La somma delle altezze delle barre è 100% o 60%?
- L'altezza della barra “pessimo” è zero o uno?
- Perché le barre sono blu?

Potreste argomentare che in questo esempio i valori precisi non sono importanti per il grafico. La cosa importante è il “messaggio”, cioè che ci sono più valutazioni “ottimo” e “buono” che valutazioni “scadente” o “pessimo”. Ma per trasmettere questo messaggio o si usa una frase, oppure si disegna un diagramma che convogli questo messaggio in modo più chiaro, come nella figura 9.10.

Il grafico della figura 9.10 ha pressappoco la stessa densità di informazione della tabella (ha le stesse dimensioni e vi sono riportati gli stessi numeri). In più uno può direttamente “vedere” che ci sono più valutazioni “buone” e “ottime” che non “scadenti”. Uno può anche “vedere” che il numero delle persone che non hanno formulato valutazioni di nessun tipo, non restituendo il questionario, non è trascurabile, e questo è un fatto abbastanza comune quando si distribuiscono questionari di valutazione.

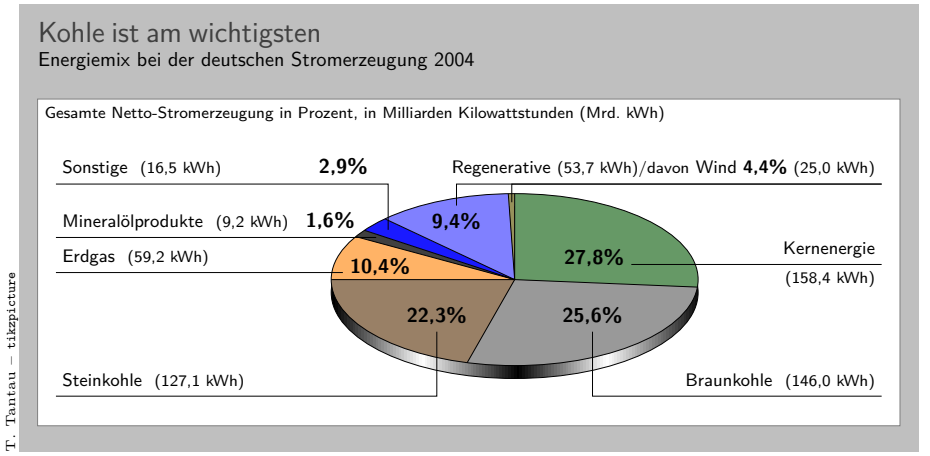


Figura 9.11: Esempio di diagramma a torta per confrontare le fonti di produzione dell'energia elettrica in Germania nel 2004. Regenerative: rinnovabile; Wind: eolica; Kernenergie: nucleare; Braunkhole: lignite; Steinkhole: litantrace; Erdgas: metano naturale; Mineralölprodukte: petrolio e derivati; Sonstige: altre fonti.

Anche i diagrammi spaziali spesso non sono efficaci. Esaminiamo un altro esempio nella figura 9.11 che io ho ridisegnato traendolo dalla rivista *Die Zeit*, 4 giugno 2005. Questo diagramma della figura 9.11 è stato ridisegnato con Ti*k*Z, ma è praticamente identico all'originale.

A prima vista esso sembra ben fatto, completo e pieno di informazioni, ma ci sono diverse cose che non sono a posto.

- Il grafico è tridimensionale. Tuttavia le sfumature di colore non aggiungono nulla dal punto di vista informativo, al più distraggono.

- In un grafico 3-D le dimensioni relative sono molto distorte. Per esempio, l'area occupata dalla zona grigia "Braunkhole" è maggiore dell'area occupata dalla zona verde della "Kernenergie", *malgrado il fatto che la percentuale dell'energia prodotta con il Braunkhole sia inferiore alla percentuale di quella nucleare.*

- La distorsione della grafica 3-D diventa ancora peggiore per le piccole aree. L'area dell'energia "Regenerative" risulta più grande di quella occupata dall'area destinata all' "Erdgas". L'area del "Wind" è appena minore di quella "Mineralölprodukte" *anche se la percentuale del Wind è quasi tre volte maggiore di quella del Mineralölprodukte.*

  In quest'ultimo caso le dimensioni diverse sono solo in parte dovute alla distorsione 3-D. I disegnatori del grafico originale hanno anche fatto la fetta "Wind" troppo piccola anche tenendo conto della distorsione. (Si confrontino in generale le dimensioni del "Wind" con quelle del "Regenerative".)

- Stando alla didascalia di questo grafico, esso dovrebbe darci l'informazione che in Germania il carbone è stata la sorgente di energia più importante nel 2004. Tralasciando le forti distorsioni prodotte dall'impostazione 3-D, ci vuole della buona volontà per percepire il messaggio.

  Il carbone come fonte di energia è diviso in due fette, una per il "Braunkhole" e l'altra per il "Steinkhole" (due diversi tipi di carbone). Quando si uniscano le due fette, solo allora si vede che la metà inferiore della torta è presa dal carbone.

  Inoltre le due aree per i due differenti tipi di carbone non sono per niente collegate visualmente. Invece sono usati due colori differenti e le rispettive legende sono ai lati opposti del disegno. A confronto, "Regenerative" e "Wind" sono collegate molto strettamente.

- Il codice dei colori del grafico non segue nessuna regola logica. Perché l'energia nucleare è segnata in verde? Perché l'energia rinnovabile è azzurra e quella derivata da altre fonti è blu? Sembra addirittura una barzelletta che l'area del Braunkhole (letteralmente del carbone marrone) sia grigio pietra, mentre l'area dello Steinkhole (letteralmente carbone di pietra) sia marrone.

- L'area con il colore più chiaro è destinata all' "Erdgas". Quest'area emerge dal diagramma con più forza, sebbene secondo questo diagramma l' "Erdgas" non sia veramente importante.

Edward Tufte chiama i grafici come quello della figura 9.11 "grafici spazzatura".

Ecco allora alcune raccomandazioni per aiutarvi a non produrre grafici spazzatura:

- Non fate diagrammi a torta tridimensionali: questi sono *spazzatura*.

- Esaminate se non sia meglio preferire una tabella a un diagramma a torta.

- Non usate i colori a caso; usateli invece per dirigere l'attenzione del lettore e per raggruppare gli oggetti.

- Non usate tessiture di sfondo come le tessiture a graticcio o di linee diagonali; usate piuttosto i colori. Le tessiture distraggono. Le tessiture di sfondo nei diagrammi informativi sono *spazzatura*.[8]

[8] Dovendo stampare in bianco e nero e tonalità di grigio, usate preferibilmente sfondi di diverse tonalità di grigio piuttosto che tessiture di vario genere. Fate attenzione a non scegliere grigi troppo chiari o troppo scuri perché in stampa potrebbero apparire decisamente bianchi o neri. Nello stesso tempo fra grigi adiacenti state attenti ad usare grigi contrastanti o grigi concordanti a seconda della contestualizzazione delle aree da rappresentare.

9.8.8 Attenzione e distrazione

Prendete il vostro romanzo preferito ed esaminate una pagina tipica. Noterete che la pagina è molto uniforme. Non c'è nulla che possa distrarre il lettore mentre legge. Non ci sono grandi titoli, non c'è testo composto in neretto, non ci sono spazi bianchi di grandi dimensioni. Invece se l'autore vuole evidenziare qualcosa lo fa componendolo in corsivo. Il corsivo si adatta benissimo al font usato per il testo principale, verosimilmente il tondo, e a una certa distanza non riuscireste a dire se la pagina contiene lettere in corsivo, mentre notereste subito se ci fosse anche una sola una parola composta in nero. Il motivo per il quale i romanzi sono scritti così segue il paradigma seguente: "Evita le distrazioni".

La buona tipografia (come anche la buona organizzazione) è qualcosa che *non* si nota. Il compito della tipografia è quello di rendere il più possibile comoda la lettura del testo e quindi del recepimento dell'informazione che esso contiene. Per un romanzo i lettori recepiscono l'informazione leggendo il testo una riga dopo l'altra, come se stessero ascoltando qualcuno che racconta una storia. In questa situazione qualunque cosa ci sia sulla pagina, che distrae l'occhio dall'andare rapidamente e uniformemente da una riga alla successiva, rende la lettura del testo meno agevole.

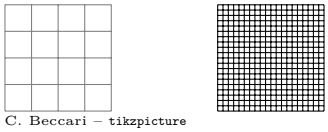
Ora prendete la vostra rivista preferita o il vostro quotidiano ed esaminate una pagina tipica. Noterete che la pagina contiene un gran "movimento". I font sono usati con corpi differenti e con strutture compositive diverse; il testo è organizzato in colonne strette, spessissimo interrotte da immagini. I motivi per i quali le riviste e i quotidiani sono composti in questo modo segue un altro paradigma: "Guida l'attenzione".

I lettori non leggono una rivista come un romanzo. Invece di leggere una rivista una riga dopo l'altra, essi leggono i titoli o dei brevi riassunti (sottotitoli) per verificare se sono interessati a leggere l'intero articolo. Nel momento, però, in cui il lettore ha deciso che vuole leggere l'articolo, egli non vuole più essere distratto ed è per questo che gli articoli sono scritti proprio come i romanzi.

I due principi "evita le distrazioni" e "guida l'attenzione" si applicano anche alla grafica. Quando disegnate un grafico dovete eliminare ogni cosa che "distragga l'occhio". Nello stesso tempo dovete cercare attivamente di guidare il lettore "attraverso il grafico" usando i font, i colori e gli spessori delle linee per evidenziare le varie parti.

Qui c'è una lista, assolutamente non completa, di cose che possono distrarre il lettore:

- I forti contrasti attraggono sempre l'occhio. Per esempio, si considerino le seguenti due griglie:



Sebbene la griglia di sinistra appaia per prima quando noi leggiamo la pagina nella direzione solita da sinistra a destra, è verosimile che la griglia di destra sia percepita per prima; il contrasto bianco-nero è maggiore di quello che si trova nel contrasto grigio-bianco[9]. Inoltre nella griglia di destra ci sono più punti di cambiamento di contrasto che ne aumentano l'effetto.

Oggetti come le griglie e, in generale, le linee delle coordinate, non dovrebbero di solito attrarre l'attenzione del lettore e quindi dovrebbero essere disegnate con un debole contrasto rispetto allo sfondo. Inoltre una griglia a maglie larghe distrae di meno di una griglia a maglie strette.

- Le linee tratteggiate creano molti punti in cui c'è un contrasto bianco-nero. Le linee tratteggiate o punteggiate in generale distraggono molto.

  Non usate linee con differenti tratteggi per differenziare curve diverse. Si perdono dei dati in questo modo (a meno che lo stacco fra i trattini e/o i puntini non sia molto piccolo) e l'occhio non riesce bene a raggruppare gli oggetti sulla base del tratteggio. L'occhio riesce *molto meglio* a raggruppare gli oggetti sulla base dei colori.

  Stampando in bianco e nero talvolta è necessario violare questa raccomandazione: progettando bene i diagrammi è possibile usare i tratteggi, come è possibile usare tonalità di grigio. Tuttavia se le dimensioni e le distanze delle linee da tracciare (mai più di quattro sullo stesso diagramma) lo consentono, si possono tracciare le varie curve con linee continue differenziandole mediante l'apposizione di triangolini, quadratini, cerchietti, eccetera, su ciascuna curva, specialmente se collocati in corrispondenza dei punti (nodi) da tracciare lungo ciascuna curva.

- Inserire degli sfondi con linee diagonali, verticali o orizzontali o anche semplicemente con dei puntini, risulta quasi sempre una distrazione e tipicamente non serve a niente.

- Piccole immagini (clip arts) possono essere simpatiche ma di solito distraggono.

9.8.9 Commenti

Till Tantau con molta modestia afferma che ha preso queste raccomandazioni e queste linee guida da libri famosi e autorevoli, in particolare da quelli di Edward Tufte. Egli spera che queste raccomandazioni non

[9] Se state esaminando queste due griglie sullo schermo, tenete conto che le linee della griglia di sinistra sono tracciate in colore grigio, il grigio di default con 75% di bianco e 25% di nero; la griglia di destra è disegnata in nero. La qualità dello schermo e la sua tecnologia, nonché la sua densità di pixel potrebbe rendere difficile la percezione di queste differenze.

rappresentino le sue personali preferenze, ma siano il meglio che è riuscito a distillare da quei libri.

In alcuni casi le linee guida indicate sembrano un po' troppo cogenti o, benché perfettamente comprensibili, lasciano il dubbio di essere difficili da applicare.

Certo, quello che si deduce da queste righe è che, come la composizione tipografica è un'arte che va appresa e coltivata, così la grafica richiede non poca dedizione e studio oltre alla abilità personale del disegnatore.

Oggi dovrebbe essere più facile di ieri trovare un buon grafico che possa fare i nostri disegni; ma anche il miglior grafico di questo mondo non farà dei bei disegni se noi stessi non siamo capaci di informarlo di che cosa il disegno deve trasmettere al lettore; è molto più facile farlo con lo scritto che con il disegno.

Tuttavia quelli di noi che devono produrre documenti scientifici con un certa dose di grafici, diagrammi e pittogrammi è bene che imparino queste linee guida (compreso il fatto che le linee guida possono anche non essere rispettate se lo si fa con cognizione di causa). Esistono delle norme UNI [59] che danno altre linee guida (non necessariamente in contrasto con quelle qui esposte, forse solo esposte con parole diverse) che chi deve produrre disegni tecnici dovrebbe sempre osservare. Debbo dire che gli esempi riportati nella versione cartacea di queste norme sono bellissimi e, per quello che si può dire dal risultato visivo, sembrano perfettamente coerenti con le linee guida qui esposte.

È bene sottolineare che le norme di disegno statunitensi sono molto diverse da quelle europee; un disegno fatto seguendo quelle norme si distingue a colpo d'occhio; ma si distingue a colpo d'occhio anche un disegno costruito senza seguire nessuna norma; di solito il risultato non è affatto bello.

Capitolo 10

LaTeX: l'importazione di figure esterne

10.1 Introduzione

Come accennato nel capitolo precedente, capita di dover importare disegni o immagini prodotte con strumenti esterni. Per quel che riguarda i disegni al tratto si è già spiegato quali vantaggi ci siano ad usare i programmi di disegno nativi del sistema TeX. Tuttavia può succedere di dover importare disegni o immagini prodotti all'esterno.

Qui ci concentreremo sulle immagini, sostanzialmente le fotografie o altri disegni a mezze tinte non ottenibili da un semplice programma specializzato, per esempio, nel tracciare diagrammi a due o a tre dimensioni.

Il pacchetto *pgf* gestisce anche l'importazione delle immagini esterne, ma lo strumento principe per operare in questa direzione è senz'altro il pacchetto *graphicx* che da anni costituisce l'interfaccia di riferimento.

Prima però di addentrarci nell'uso di quel pacchetto bisogna fermarsi un attimo per meditare sulla moltitudine di formati grafici che rendono le cose meno semplici di quanto potrebbero essere.

10.2 I formati grafici

I formati grafici si distinguono sostanzialmente in formati vettoriali e in formati a matrici di pixel. Essi hanno caratteristiche diverse e per certe applicazioni vanno meglio i primi, mentre per certe altre vanno meglio i secondi.

10.2.1 I formati vettoriali

I formati vettoriali sono descritti mediante una specie di programma che contiene le istruzioni per tracciare i contorni delle aree dove il colore è sostanzialmente

costante; questi contorni sono costituiti dalla conoscenza delle serie di punti per i quali devono passare le curve di Bézier che li descrivono. Questi formati sono usatissimi per tracciare anche quei particolari disegni costituiti dai caratteri dei vari font; questi infatti devono potersi disegnare rapidamente, ma, specialmente se devono essere presentati sullo schermo, devono essere scalabili a piacere senza che l'ingrandimento o il rimpicciolimento producano difetti degni di nota.

È chiaro che se il disegno è costituito dai contorni delle aree dello stesso colore, il problema del cambiamento di scala ha una soluzione banale, in quanto basta moltiplicare le coordinate dei punti che descrivono il contorno per il fattore di scala; per il resto non cambia praticamente nulla. In realtà, quando si rimpicciolisce un disegno che contiene delle linee sottili, queste diventano ancora più sottili e, per esempio, sullo schermo, che ha una densità di puntini luminosi abbastanza bassa, da 70 a 100 puntini al pollice, queste linee diventerebbero troppo sottili per essere rappresentate; l'algoritmo di rappresentazione deve contenere quindi dei suggerimenti (*hints*) per produrre un effetto finale che non lasci sparire le linee troppo sottili.

In questo testo l'argomento diventerebbe troppo tecnico e non si insiste oltre; tuttavia vale la pena di ricordare che i formati grafici vettoriali più noti e frequenti sono il formato PostScript (estensione del file `.ps`) e il suo parente stretto Encapsulated PostScript (estensione `.eps`); oggi sta diventando sempre più frequente il formato Scalable Vector Graphics (estensione `.svg`), usato specialmente per le applicazioni Web.

C'è anche il formato di uscita del programma METAPOST che, partendo da un file sorgente simile a quanto si potrebbe scrivere con il programma di creazione dei font creato da Donald E. Knuth, produce un'uscita in una specie di linguaggio PostScript ridotto e semplificato. Il programma è molto utile per produrre disegni al tratto, che vanno dai disegni ai diagrammi, ma l'unico difetto è che il linguaggio non è dei più semplici e quindi il programma è poco usato, comunque meno di quanto meriterebbe.

A stretto rigore il formato Portable Document Format (estensione `.pdf`) sarebbe un formato vettoriale e sotto molti aspetti lo è. Tuttavia il file è un file con il codice compresso (zippato, secondo una diffusa terminologia), ma il file decompresso è sostanzialmente un sotto insieme del formato PostScript, quindi sostanzialmente vettoriale. Il suo tipo di compressione richiede programmi speciali per la visualizzazione e stampa, destinati specialmente a rendere la pagina sotto forma grafica.

10.2.2 I formati diversi da quelli vettoriali

Le fotografie sono, oggi più spesso di ieri, in formato JPEG (estensione `.jpg`). Tuttavia i formati di matrici di pixel sono diffusissimi, e le estensioni vanno da `.bmp` a `.wmf`, da `.tiff` a `.gif` e a Portable Network Graphics (estensione `.png`); questa è solo una piccola elencazione di formati grafici, perché esistono dei formati specifici per certi particolari apparecchi fotografici digitali o per certi codici di colore.

Il vantaggio dei formati a matrici di pixel è che i programmi per la loro presentazione sono particolarmente rapidi a produrli in forma visibile per gli umani. Tuttavia hanno un paio di difetti tutt'altro che trascurabili.

Il primo difetto è che un'immagine di pochi pixel non può essere ingrandita su un'area che contenga più pixel, perché essa non contiene abbastanza informazione per colorare correttamente i pixel della riproduzione. Un'immagine quadrata di 100 pixel per 100 pixel può essere rappresentata abbastanza bene sullo schermo di un PC ed apparirà un'immagine di circa 30 mm per 30 mm. Se si vuol vedere a pieno schermo la stessa immagine, cioè su un'area di 800 pixel per 640 pixel, si capisce bene che il meglio che si possa fare è di rappresentare i pixel dell'immagine come quadrati di circa 8 pixel per 8 pixel, cioè sgranando l'immagine, cosicché appare come un mosaico di quadrati tutt'altro che piccoli, con la conseguenza che i contorni non sono più nitidi. La situazione peggiora notevolmente sulla carta dove il quadrato di 100 pixel per 100 pixel copre un quadratino di circa 8 mm per 8 mm stampando con una stampante con una densità di 300 puntini al pollice e di circa 4 mm per 4 mm stampando a 600 puntini al pollice; se si volesse ingrandire l'immagine come la si vede sullo schermo, l'effetto mosaico sarebbe talmente importante che difficilmente si riuscirebbe a riconoscere l'immagine.

È vero, esistono programmi che consentono di eseguire un filtraggio bidimensionale in modo da ridurre l'effetto mosaico, ma questo si paga con una minore definizione dei contorni, cosa che spesso rappresenta il male minore, tenuto conto della poca informazione contenuta nel file di partenza; non si pensi di ottenere quei meravigliosi effetti che vengono mostrati in televisione, dove la polizia scientifica ingrandisce le immagini delle videocamere di videosorveglianza (che notoriamente riprendono le immagini con una definizione piuttosto bassa) fino a leggere numeri di targa o a riconoscere i lineamenti del colpevole!

La situazione è un poco migliore se si vuole ridurre l'immagine su un'area che contenga un po' meno pixel, ma il tutto è affidato alla qualità del tipo di rappresentazione o di stampa. Conclusione: si preferiscano sempre i formati vettoriali che possono essere ingranditi e rimpiccioliti a piacere. Tuttavia talvolta "bisogna fare di necessità virtù".

Se a questo si aggiunge il secondo inconveniente dovuto al fatto che alcuni formati sono compressi in una maniera che sfrutta la ridondanza delle immagini, ci si rende conto che un simile metodo perde parte dell'informazione e quando il file viene decompresso per la resa dell'immagine su schermo o su carta, appaiono degli artefatti che derivano dall'impossibilità di comprimere il file in modo decisivo senza perdere parte dell'informazione che l'immagine contiene. Se si tratta di una fotografia con colori sfumati e senza contorni netti, questo disturbo si nota poco, ma se si trattasse di un disegno al tratto la resa potrebbe essere inaccettabile.

Per le fotografie il formato `.jpg` va abbastanza bene ma può variare notevolmente da una fotografia all'altra. Per i disegni al tratto, se non fosse possibile avere dei disegni in formato vettoriale, allora il formato più adeguato sarebbe quello con estensione `.png`. Entrambi i formati sono compressi, ma la compressione del formato `.jpg` è una *lossy compression*, cioè con perdita di informazione,

mentre per il formato `.png` si ha una *lossless compression*, cioè senza perdita di informazione. La compressione del formato `.jpg` è più efficace di quella `png` ma a scapito della qualità dell'immagine. Il formato `.png` ha anche un vantaggio rispetto a quello `.jpg`, nel senso che gestisce le trasparenze di colore che, invece, il formato `,jpg` non può gestire.

In ogni caso, fra i formati di questo paragrafo, quello di gran lunga preferibile sarebbe il formato `.pdf`, almeno nelle circostanze in cui il file PDF è stato composto senza perdere la natura vettoriale del disegno di partenza.

Tanto per rendere più visibile il fenomeno appena descritto, la figura 10.1 mostra due diagrammi ottenuti con Excel e salvati uno in formato PNG e l'altro in formato PDF. Per entrambi si è ingrandito di circa 10 volte il quadratino evidenziato sulla parte di sinistra. Come si vede, l'effetto di quantizzazione nella figura in formato PNG rende l'ingrandimento assolutamente irriconoscibile, mentre l'ingrandimento della figura vettoriale in formato PDF è perfetto e si distinguono persino i puntini sui quali Excel ha interpolato la curva.

10.3 I formati accettabili a seconda del programma di composizione

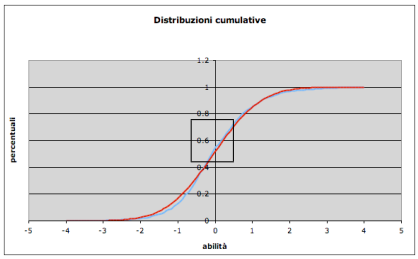
A seconda che si usi LaTeX o pdfLaTeX alcuni formati sono accettabili e altri no. Peccato. Il solo formato accettabile da tutti sarebbe il formato di uscita di METAPOST. Ma anche per questo ci vuole un poco di maquillage. L'estensione del nome di questi file è costituita da un numero progressivo, per esempio `.008` perché si trattava dell'ottavo disegno in uscita dall'elaborazione dello stesso file sorgente. Questa estensione è accettabile da LaTeX ma non è accettabile da pdfLaTeX. Perciò è necessario cambiare l'estensione di tutti i file di uscita da METAPOST, aggiungendo l'estensione `.mps` o meglio cambiandone il nome, per esempio da `mypicture.008` a `mypicture-008.mps`. Ciò fatto la cosa va bene sia per LaTeX sia per pdfLaTeX, ma oltre al modo manuale esiste, dalla versione 1.000 di METAPOST, un modo che sfrutta un meccanismo di denominazione dei file per *modelli*.

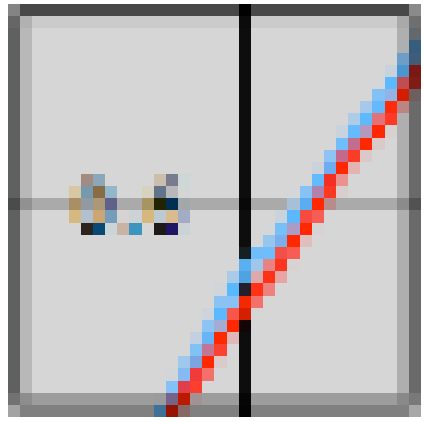
Un modello di nome è una stringa racchiusa tra doppi apici che include *codici segnaposto* tra cui `%j` per il nome del file sorgente e `%3c` per il numero di figura con formato fisso a tre caratteri. Otteniamo quindi lo stesso effetto descritto dell'esempio precedente scrivendo l'istruzione di preambolo del file sorgente `.mp`[1]

```
outputtemplate := "%j-%3c.mps"
```

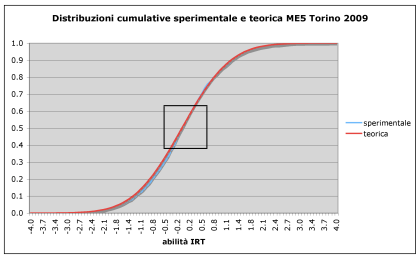
Una alternativa legata all'uso del pacchetto *graphicx* potrebbe essere quella di definire una nuova regola di elaborazione dei file grafici; se si usa pdfLaTeX, si può usare il comando `\DeclareGraphicsRule` e specificare nel preambolo

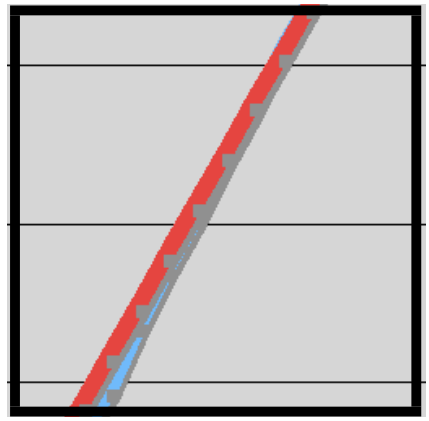
[1] Il comando METAPOST `outputtemplate` ha sostituito dalla versione 1.200 il comando `filenametemplate` che quindi oggi è classificata come deprecata. Notizie complete sui codici segnaposto si trovano nel manuale a cui si rimanda.





(*a*)





(*b*)

C.Beccari – picture, Microsoft Excel, Mac OS X Preview

Figura 10.1: Confronto sugli effetti dell'ingrandimento su due figure simili. (*a*) Immagine in formato bitmapped PNG; (*b*) in formato vettoriale PDF; la parte ingrandita a destra corrisponde al riquadro evidenziato nella parte di sinistra.

```
\DeclareGraphicsRule{*}{mps}{}{}
```

Con questo si dichiara che qualunque altro formato (asterisco), diverso da quelli per i quali esiste una apposita regola, debba venire trattato come un file ottenuto da METAPOST. Questo è un escamotage che presenta qualche inconveniente, ma se si è sicuri di usare solo file ottenuti da METAPOST, non dovrebbe presentarsi nessun problema.

Tuttavia bisogna stare attenti a tutti gli altri formati secondo quanto detto qui di seguito.

10.3.1 I formati accettabili da latex

latex di per sé non accetta nessun formato grafico; sono i driver DVI che accettano questo o quel formato. Il driver dvips accetta solo i formati vettoriali; quindi oltre all'uscita di METAPOST, esso accetta solo i formati `.eps` e `.ps`; il formato Encapsulated è preferibile rispetto al formato PostScript, perché sicuramente non contiene comandi PostScript che risultano incompatibili con l'inserimento delle figure all'interno di un'altra pagina PostScript. Benché il formato possa essere vettoriale, dvips non accetta assolutamente il formato `.pdf`.

Invece il driver dvipdfm accetta sia il formato d'uscita del programma METAPOST, sia i formati vettoriali PostScript `.eps` (preferibilmente) e `.ps`, sia i formati `pdf`, `.jpg` e `.png`. Per questi ultimi necessita di file ausiliari che contengano le dimensioni del rettangolo tangente (bounding box), ma la distribuzione del pacchetto contiene anche un programmino ebb che estrae queste informazioni e le trascrive in file adeguati per l'uso che dvipdfm ne deve fare. Per i file PostScript in formato `.eps` e `.ps` esso ricorre automaticamente al processo di conversione attraverso l'invocazione del programma gs per ottenerne la versione `.pdf`. Questo ricorso al programma esterno è necessario perché dvipdfm è capace di interpretare codice PostScript sufficientemente semplice (per esempio quello contenuto nel file di uscita di METAPOST), ma non è in grado di interpretare codice PostScript arbitrariamente complesso. Questo è il motivo per il quale dvipdfm non è in grado di trattare file DVI che contengano le istruzioni PostScript generate dal pacchetto *PSTricks*.

Si segnala l'esistenza del programma dvipdfmx, un'estensione di dvipdfm, che dovrebbe presentare una maggiore funzionalità. Unito ad altri pacchetti di estensione, come per esempio *pdfcomment* consente di creare le *annotazioni*, una caratteristica del formato PDF, che non sarebbe nativamente realizzabile con il semplice latex. Tuttavia, salvo alcune limitazioni importanti, come l'impossibilità di convertire direttamente file che contengano codice PostScript avanzato, come quando si usa *PSTricks*, sembra che questo programma esteso sia il più efficace nel trasformare un file DVI nel corrispondente file PDF in termini di ingombro di memoria.

A questo proposito merita di segnalare una tabellina, qui riprodotta in modo semplificato, generata da Péter Szabó e pubblicata negli atti della EuroTEX

Metodo di compilazione	Ingombro in byte
pdftex	2 283 510
tex + dvipdfm	2 269 821
tex + dvipdfmx	2 007 012
tex + dvips + ps2pdf	3 485 081

Tabella 10.1: Dimensioni in byte dei file PDF ottenuti compilando il TEXbook con diversi metodi

Conference del 2009, dove sono riportati gli ingombri in byte dei file PDF ottenuti con diversi procedimenti, tabella 10.1.

Infine si può mettere in evidenza un modo "pulito" per ottenere in uscita un file PDF direttamente da un file predisposto per essere compilato con latex e faccia uso dei comandi avanzati generati dal pacchetto *PSTriks*. Si tratta di usare il programma di compilazione ps4pdf; lo si può lanciare da linea di comando, ma si possono configurare diversi *shell editor* perché lo annoverino fra i motori di composizione che possono lanciare. Per TEXShop per esempio, basta aggiungere agli altri file `.engine` di cui già dispone nella cartella `~/Libreria/TeXShop/Engines/`, il file `ps4pdf.engine` contenente quanto segue:

```
#!/bin/tcsh

set path= ($path /usr/texbin /usr/local/bin)
ps4pdf  -file-line-error -shell-escape -synctex=1 "$1"
```

A questo file bisogna impostare il bit per poterlo eseguire dando da terminale il comando:

```
\chmod 755 ps4pdf.engine
```

Ciò fatto fra i motori di composizione che compaiono nel menù di TEXShop compare ora anche ps4pdf che può essere usato in alternativa agli altri programmi di composizione. Una impostazione del tutto simile si può fare per molti altri *shell editor*, naturalmente con le varianti specifiche di ciascuno di essi.

Si segnala che questo programma, pur essendo uno script che esegue in background diversi altri programmi del sistema TEX, non si limita a compilare il file DVI e a convertirlo in formato PDF con le procedure descritte sopra, ma estrae le parti scritte con i comandi e gli ambienti di *PSTriks*, le compila, le trasforma in formato PDF, le scontorna, e infine le rende disponibili ad una vera compilazione con pdflatex, il che implica che si possono avere nel file finale tutte le prestazioni di una file scritto direttamente con il linguaggio PDF, quindi, in particolare, si possono mantenere tutti i vantaggi della microgiustificazione. Il file sorgente del documento, infatti, può anche usare, per esempio, il pacchetto *pdfpages*, di Andreas Matthias, che si ringrazia per la segnalazione; questo pacchetto può funzionare solo se il motore di composizione gestisce i comandi avanzati del linguaggio PDF.

10.3.2 I formati accettabili da pdflatex

pdfLaTeX, oltre al formato di uscita di METAPOST e al formato `.pdf`, per ovvi motivi di compatibilità, accetta in entrata anche i formati `.jpg` e `.png`. La versione distribuita con TeX Live a partire dal 2009 è stata estesa per convertire automaticamente il formato `.eps` in formato `.pdf`. Dal 2010 TeX Live è attrezzato per eseguire direttamente la conversione senza bisogno di installare o invocare nessun ulteriore pacchetto di estensione.

La figura 10.2 contiene infatti l'inclusione del file grafico `tiger.eps`, che fa parte dei file di esempio della distribuzione di GhostScript, convertito al volo durante l'esecuzione del programma pdflatex con cui è stato composto questo documento. Durante l'esecuzione si osserva una breve pausa, una tantum, durante la quale l'interprete pdftex si ferma un istante per lanciare un programma di sistema per convertire il formato `.eps` dell'immagine nel formato `.pdf`. Questa operazione è eseguita *una tantum*, nel senso che se il file convertito esiste già, l'interprete incorpora l'immagine convertita direttamente; se però l'immagine `.eps` è stata aggiornata con una versione più recente, la conversione viene eseguita di nuovo. I comandi per generare la figura 10.2[2] sono i seguenti:

```
\begin{figure}\centering
\includegraphics[width=.7\textwidth]{tiger.eps}
\caption{Inclusione del file \texttt{tiger.eps} da parte di
    \prog{pdflatex}}
\label{fig:tiger}
\end{figure}
```

Altrove e in versioni precedenti di questa Introduzione si è consigliato di non usare l'estensione dei file grafici; i motivi in parte erano e sono tuttora spiegati nella documentazione dei file *epsconversion* e *epstopdf*; ma dal 2010 ritengo che non si debba più seguire questo consiglio. Prima del 2010, infatti, nella stessa cartella potevano esistere sia un file `.eps` sia la sua conversione in un formato accettabile da pdflatex; il nome dei due file poteva essere lo stesso ma ne era diversa l'estensione. In questo modo, se non si specificava l'estensione, lo stesso file sorgente poteva essere elaborato sia da latex, che avrebbe letto e incluso solo i file `.eps`, sia da pdflatex, che avrebbe letto solo uno degli altri formati.

Ora, con la capacità di pdflatex di gestire i file `.eps` come se fossero nativi della sua natura, due file con lo stesso nome e diversa estensione possono non rappresentare più la stessa figura memorizzata in formati diversi, ma potrebbero rappresentare due figure distinte. Si commenta da sola l'inopportunità di distinguere le figure solo sulla base dell'estensione, ma non del nome; ma proprio per evitare confusioni e rendere più spedita la ricerca della figura da includere, ritengo che sia meglio specificare sempre anche l'estensione del file grafico che contiene la figura.

[2] La figura `tiger.eps` viene distribuita insieme al programma `gs`.

Figura 10.2: Inclusione del file `tiger.eps` da parte di pdflatex

Si veda anche il paragrafo precedente in merito all'uso di ps4pdf, che consente di usare pdflatex anche quando il file da compilare usa il pacchetto *PSTricks* con i suoi comandi avanzati in linguaggio PostScript.

Dal 2012 è disponibile anche il pacchetto *svg* che permette di includere le immagini nel formato `.svg`; dal gennaio 2013 il pacchetto è disponibile nella distribuzione TEX Live 2012 ma, siccome si appoggia a programmi esterni, richiede l'attivazione indiscriminata della opzione *shell-escape*) (o *enable-write18*) per poter agire su programmi esterni che non sono disponibili per tutte le piattaforme, per cui funziona solo sulle macchine Linux e Mac. Siccome il programma inkscape è disponibile per tutte le piattaforme, il pacchetto menzionato qui, che non è ancora operativo sulle macchine Windows, può venire ignorato su queste piattaforme purché si abbia la pazienza di fare a mano quello che il pacchetto *svg* è in grado di fare da solo. In sostanza sia usando il pacchetto, sia procedendo a mano si producono due file, uno in formato PDF che contiene l'immagine senza legende, e un secondo file con il mark up LATEX contenente le istruzioni per comporre e collocare al punto giusto le legende. Il tutto risulta molto comodo per eseguire disegni vettoriali con una interfaccia grafica interattiva, inkscape, che in certe circostanze può risultare di più comodo impiego rispetto all'uso dei programmi di disegno indicati in questa guida.

10.4 Conversione dei formati

Ci troviamo forse in un vicolo cieco? Per fortuna no. Basta poter convertire da un formato all'altro. Esistono diversi programmi per farlo e qui se ne elencano alcuni; la maggior parte sono disponibili per tutte le piattaforme. Inoltre, servendosi della distribuzione 2010 o successiva di TeX Live, molte delle conversioni elencate qui sotto diventano superflue quando si usi pdflatex; esse restano necessarie quando si debbano includere figure, per esempio in formato `.bmp` o in altri formati derivanti da macchine fotografiche particolari o da sistemi operativi diversi da quello col quale si sta lavorando.

ghostscript e, meglio ancora, la sua interfaccia grafica ghostview (conosciuta anche col nome gview) permettono di trasformare una immagine `.ps` oppure `.eps` in immagine `.pdf` e viceversa. All'occorrenza il programma è capace di trasformare un file `.ps` in un file `.eps` assicurando che il file finale non contenga istruzioni PostScript incompatibili con la possibilità di inclusione in un altro file.

epstopdf epspdf e ps2pdf, o altri simili applicativi da linea di comando, eseguono le analoghe trasformazioni senza interfaccia grafica; spesso hanno il vantaggio di corredare il file di uscita dell'informazione corretta in merito al 'bounding box', cioè alle dimensioni del rettangolo circoscritto all'immagine effettiva. Con la distribuzione 2010 di TeX Live non è necessario ricorrere a programmi esterni, nel senso che ci pensa direttamente pdflatex a eseguire la conversione senza che l'utente debba preoccuparsene. Bisogna ricordare che la conversione dal formato `.eps` al formato `.pdf` talvolta *non* comporta l'inclusione dei font eventualmente usati nel file di partenza; questo fatto può portare a certi inconvenienti, quando il file `.pdf` viene incorporato in un file principale che verrà letto su un'altra macchina che non dispone degli stessi font; oppure rende il file principale incompatibile con il formato archiviabile (vedi il capitolo 22).

eps2pdf è una interfaccia grafica disponibile solo per macchine Windows che esegue sostanzialmente lo stesso tipo di trasformazione di epstopdf, ma con il vantaggio dell'interfaccia grafica.

eps2png e eps2jpg sono programmi da linea di comando per macchine Linux/UNIX; trasformano le immagini vettoriali in formato `.eps` in immagini a matrici di punti diversamente compresse nei formati `.png` oppure `.jpg`. In generale sarebbe desiderabile evitare questi tipi di conversione, ma è opportuno limitarsi alle conversioni eseguibili con programmi descritti nei punti precedenti che conservano la qualità vettoriale dell'immagine. Tuttavia, in vista di produrre file archiviabili, talvolta questa potrebbe essere una soluzione per trasformare file `.pdf` incompatibili con l'archiviabilità in file compatibili, in quanto i "disegni" dei font vengono trasformati in

"disegni" a matrici di pixel; certo se sulla macchina dove si esegue la trasformazione mancano i font scalabili, anche questo tipo di trasformazione diventa inutile.

pdftops esegue da linea di comando la conversione dal formato `.pdf` al formato `.ps`. Anche se il formato `.pdf` è sempre da preferire, talvolta può rendersi necessario ricorrere a questo tipo di trasformazione.

jpegtops trasforma un file `.jpg` in formato `.eps`. Si tratta di un applicativo da linea di comando. Assicura che il bounding box sia specificato correttamente. In realtà questo programma si limita ad avviluppare il codice compresso dell'immagine a matrice di pixel dentro un involucro PostScript, che contiene l'informazione corretta del 'bounding box', ma per il resto si limita a definire dove inizia e dove finisce il codice dell'immagine compressa.

mptopdf esegue la trasformazione di un file prodotto con METAPOST in un file `.pdf`. Questa trasformazione in effetti non sarebbe necessaria se si dispone di un sistema TEX moderno e completo; tuttavia se si dovesse mandare la figura a qualcun altro di cui non si conoscono le dotazioni informatiche, è meglio mandargliela in formato PDF piuttosto che nel formato di uscita di METAPOST, perché il destinatario potrebbe non avere familiarità con questo formato, oppure potrebbe mancare del software necessario per visualizzarla o per includerla nei suoi documenti.

Vale la pena di segnalare che lo *shell editor* dei sistemi Mac OS X, TEXShop, consente anche di eseguire la compilazione di un file METAPOST, ma non si limita ad ottenere il file in codice PostScript, bensì provvede ad eseguire direttamente il programma mptopdf cosicché il risultato finale è direttamente il file PDF contenente il disegno perfettamente ritagliato al rettangolo tangente. Anche alcuni altri *shell editor*, certamente lo shareware WinEdt per macchine Windows, consentono di eseguire la compilazione dei disegni METAPOST; forse qualcuno consente anche la trasformazione automatica nel formato PDF.

Paint e i programmi simili aprono i file bitmapped e li trasformano in qualunque altro file a matrici di pixel, sia `.png` sia `.jpg`; quindi per gli altri formati bitmapped non dovrebbero esserci problemi per ottenere formati compatibili con i programmi di nostro interesse.

gimp si trova per tutte le piattaforme; è un formidabile programma di editing grafico che permette di eseguire tutte le trasformazioni dell'immagine e la loro conversione in un qualunque formato a matrici di pixel; riesce a farlo anche con input vettoriali, ma salva solo in formati a matrici di pixel. È un programma fortemente consigliabile per chiunque e su qualunque piattaforma. Sarebbe però consigliabile non usarlo per convertire formati vettoriali in formati a matrici di pixel, limitandosi alle altre trasformazioni e all'editing degli altri formati.

ImageMagik è un altro noto programma di editing grafico; esso mette a disposizione dell'utente il comando convert che permette di eseguire rapidamente la conversione da un formato grafico all'altro; viene consigliato (ed effettivamente viene installato automaticamente insieme alla distribuzione MacTeX del sistema TeX sulle macchine Macintosh) e, per esempio, viene suggerito di definire la regola grafica

```
\DeclareGraphicsRule{.tif}{png}{.png}%
{‘convert #1 ‘basename #1 .tif‘.png}
```

per convertire le immagini `.tif` in immagini `.png`. Questo metodo, in realtà, può venire usato su qualunque piattaforma e richiede l'abilitazione all'esecuzione di programmi esterni mediante l'opzione della linea di comando `-shell-escape`, della cui 'pericolosità' si è già parlato.

GraphicConverter è un programma shareware per il sistema operativo Mac OS X; svolge più o meno le stesse funzioni di *ImageMagik*; se si paga il prezzo del prodotto, le prestazioni aumentano considerevolmente. Il numero di formati in ingresso e in uscita sembra essere molto più grande di quello del programma concorrente e per gli utenti di Mac OS X coinvolti in lavori che fanno uso di immagini dai formati molto eterogenei può essere la soluzione giusta.

10.5 Scontornare le immagini

Un aspetto che si dimentica troppo spesso è quello di scontornare le immagini. LaTeX lascia da solo gli spazi bianchi necessari attorno alle immagini; se queste a loro volta contengono altri spazi bianchi nel loro contorno, nel documento composto risulterà esserci alla fine troppo spazio bianco. Non è semplice poter scontornare le immagini, specialmente se ci si vuole attenere a programmi freeware.

gimp consente di scontornare e di correggere qualunque formato grafico ma permette di salvare solo nei formati a matrici di pixel; non è quindi il caso di servirsi di gimp per scontornare immagini vettoriali.

gview permette di correggere il bounding box, cosicché di fatto si scontorna l'immagine. In ogni caso questo programma permette di visualizzare sullo schermo il rettangolo tangente, cosicché diventa facile spostare il cursore a croce sul vertice inferiore sinistro e successivamente sul vertice superiore destro per rilevare nell'apposito spazio della cornice inferiore del programma le coordinate di questi due vertici; queste coordinate potranno essere usate come quelle del 'bounding box' quando si usa il comando `\includegraphics` per includere l'immagine.

Adobe Acrobat non è freeware, ma permette di eseguire numerose azioni sui file e sulle immagini in formato PDF. Si possono per esempio estrarre delle pagine contenenti immagini da un file PDF e poi si possono estrarre, scontornandole, solo le immagini che interessano andando a filo dell'immagine con una comoda interfaccia grafica. È possibile anche estrarre immagini in formato PostScript. L'unico difetto potrebbe essere il costo, visto che il programma è commerciale; il programma gratuito Adobe Reader permette di eseguire solamente la lettura ma non consente nessuna operazione di editing grafico.

Preview è un applicativo del sistema operativo Mac OS X[3]; serve a molte funzioni, in particolare per visualizzare i file PDF, JPEG, PNG, TIFF e GIF; fra le operazioni che riesce a fare su questi file c'è anche l'operazione di *cropping*, cioè di scontornare le immagini mediante una comoda interfaccia grafica; permette anche di salvare ogni file in uno degli altri formati che sa gestire. Ovviamente è usabile solo con quel sistema operativo.

pdfcrop è un programma (o uno script, a seconda della piattaforma) che permette di scontornare molto bene le immagini in formato PDF; esso è distribuito con il sistema TEX, quindi è già disponibile sulla piattaforma di lavoro e non è necessario ricorrere ad installazioni particolari.

Ma non tutto è perduto; se non si riesce a scontornare con programmi adeguati, si può sempre operare dall'interno di LATEX o pdfLATEX.

10.6 L'importazione delle immagini

Se il lettore non si è scoraggiato con queste (apparenti) difficoltà e ha già (all'occorrenza) convertito i formati delle immagini in uno dei formati consentiti, e le ha già scontornate, allora può procedere con l'effettiva importazione delle immagini.

Basta richiamare il pacchetto *graphicx* con il solito comando

```
\usepackage{graphicx}
```

e usarne i comandi e le opzioni in modo corretto.

Nel momento i cui si deve importare una figura esterna (della quale si assume di avere il file nel formato corretto in relazione al programma usato) basta usare con la seguente sintassi il comando `\includegraphics` (generalmente, ma non necessariamente, all'interno dell'ambiente *figure*):

`\includegraphics[`⟨*lista delle chiavi*⟩`]{`⟨*file*⟩`}`

Il ⟨*file*⟩ è il nome del file contenente l'immagine *preferibilmente con l'estensione specificata*; se la figura con il formato giusto esiste, LATEX o pdfLATEX importa la

[3]Nella versione italiana di Mac OS X, questo programma si chiama Anteprima.

figura; altrimenti il programma emette un messaggio di errore ma consente di proseguire, evidentemente senza importare nulla. Ovviamente se l'estensione è specificata in modo errato, il programma non trova la figura nel formato giusto e non importa nulla. Se però due diverse figure risiedessero nella stessa cartella con lo stesso nome ed estensione diversa, il programma potrebbe trovare e importare ciò che ha trovato, ma potrebbe essere una figura che non c'entra niente. Non si metterà abbastanza in evidenza l'importanza di gestire i propri file, testuali o grafici che siano, in modo intelligente al fine di evitare simili situazioni spiacevoli.

Se invece è stata definita una regola grafica, per esempio quella mostrata per convertire le immagini dal formato `.tif` al formato `.png`, allora la prima volta che si esegue la compilazione viene ricercata anche l'estensione da convertire, viene eseguita la conversione e viene usato il file convertito; nelle eventuali compilazioni successive il programma trova direttamente il file nel formato giusto e non ha più bisogno di eseguire nessuna conversione. Questo vale anche per le figure in formato `.eps` trasformate da pdflatex stesso in file con l'estensione `.pdf`.

Per i file in formato `.svg` il pacchetto *svg* mette a disposizione il comando `\includesvg` con la stessa sintassi del comando `\includegraphics`; naturalmente la ⟨*lista delle chiavi*⟩ ammette alcune chiavi specifiche per il formato `.svg` che non possono essere usate con il ocmando `\includegraphics`; il lettore è rinviato alla documentazione del pacchetto *svg* leggibile, come al solito, con il comando da terminale `texdoc svg`.

10.6.1 Organizzare le immagini

Vale la pena di esporre un dettaglio organizzativo, che però presenta vantaggi e svantaggi e va usato con intelligenza; per la gestione delle figure non ci sono problemi se i file che le contengono sono nella stessa cartella del file sorgente da comporre; tuttavia questo modo di procedere è disordinato e alla fine la cartella contiene tante cose disparate e non si riesce più a gestire. La soluzione è quella di dedicare/creare una sotto cartella dove custodire le immagini e solo loro. Supponendo di scrivere un libro su Paperino, Topolino e Pluto, probabilmente il file sorgente si troverà nella cartella `PTP/`; in questa cartella si trova la sottocartella `foto/` (se non c'è la creiamo); in questa sottocartella si trova la fotografia `Topolino.jpg`. Allora questa foto può essere inclusa per esempio specificando il percorso relativo per raggiungerla:

```
\includegraphics{foto/Topolino}
```

Più comodo e meno suscettibile di errori è invece il procedimento di specificare il percorso (*relativo alla cartella dove si trova il file sorgente*) dell'unica o delle varie cartelle, dove si trovano le immagini, mediante il comando `\graphicspath` con la seguente sintassi:

`\graphicspath{{`⟨$path_1$⟩`}{`⟨$path_2$⟩`}...{`⟨$path_n$⟩`}}`

dove è opportuno notare le parentesi graffe esterne che racchiudono le coppie di graffe interne, ognuna delle quali specifica un diverso percorso. I comandi

```
\graphicspath{{foto/}}
...
\includegraphics{Topolino}
```

permettono di includere la foto di Topolino senza bisogno di specificarne il percorso. Se le immagini relative a Paperino si trovassero in un'altra sottocartella, per esempio `fotoPaP/`, allora basterebbe specificare

```
\graphicspath{{foto/}{fotoPaP/}}
```

per avere accessibili entrambe le sottocartelle senza bisogno di specificarle per ogni inclusione[4].

Quali sono gli inconvenienti di questo metodo? Le ragioni sono ben nascoste nelle macro che ricercano i file e nel modo di accedervi. Innanzi tutto vale la pena di sottolineare che si possono dare in successione, anche in file diversi molti comandi `\graphicspath` con una lista di percorsi formati da un solo percorso o da un certo numero di percorsi.

I successivi comandi definiscono i loro percorsi aggiornando una lista conservata dentro le viscere dei programmi in uso; e questa lista può essere anche piuttosto lunga, quando con un unico comando si specificano tutti i percorsi di ricerca per l'intero documento.

Quando la lista contiene tutti i percorsi di ricerca e `\includegraphics` deve includere un file grafico come, per esempio, `Topolino.jpg`, antepone ricorsivamente al nome di questo file i nomi dei percorsi memorizzati; appena trova il file, interrompe la ricorsione, e agisce sul file. Se non si arresta prima perché il file è stato trovato, la ricorsione è condotta fino all'ultimo elemento della lista; ognuna di queste ricerche consuma tempo sia per essere eseguita sia per generare in modo automatico la concatenazione di ciascuno dei percorsi con il nome del file; se questo non avesse l'estensione, la ricerca verrebbe ripetuta anche concatenandovi ricorsivamente tutte le estensioni valide per il programma specifico di composizione.

Se si usasse quindi `\graphicspath` con 25 path diversi per includere le foto di 25 capitoli, ognuno con le sue foto in una cartella diversa e le estensioni

[4] In verità prima del 1010 sarebbe stato possibile specificare anche dei percorsi assoluti, invece che relativi alla cartella del file principale. Tuttavia anche prima del 2010 qui si consigliava di evitare questa possibilità, specialmente quando il documento deve essere composto a più mani, su macchine diverse, con sistemi operativi diversi; in queste condizioni è preferibile evitare qualunque specificazione che sia dipendente dalla particolare macchina e dal particolare sistema operativo. La portabilità dei file `.tex` è una delle caratteristiche del sistema TEX e sarebbe quanto mai inopportuno inficiare questa particolarità. Dal 2010 non si possono più specificare percorsi assoluti, quindi il problema non si pone più. Conviene ricordare ancora che anche sulle macchine Windows dove il separatore delle cartelle è il backslash, quando si specificano i percorsi all'interno di programmi con il mark-up TEX, LATEX, eccetera, il separatore delle cartelle deve essere la barra normale e ci pensa il programma di composizione a convertirlo nel backslash quando deve aprire o salvare dei file.

possibili fossero quattro, un file con la quarta estensione, da includere nell'ultimo capitolo, il 25°, verrebbe ricercato 100 volte; la foto verrebbe trovata al centesimo tentativo, sempre che il nome fosse corretto, la foto ci fosse per davvero e con una delle estensioni accettabili, eccetera. Se invece si fosse specificato semplicemente il nome del file completo di estensione e preceduto dal suo percorso relativo in modo esplicito, il file sarebbe stato trovato con una sola ricerca, e non sarebbe stato necessario ripeterla; al massimo se il percorso o il nome o l'estensione fossero stati specificati in modo errato, il programma avrebbe emesso un messaggio d'errore e l'operatore avrebbe potuto sapere subito in che cosa consisteva l'errore e avrebbe potuto poi correggerlo adeguatamente.

Non si abbia quindi paura a conservare le immagini di ogni capitolo in cartelle diverse, subordinata alla cartella che contiene il file sorgente del capitolo; la specificazione dell'intero percorso per ciascuna figura può essere noiosa, ma rende il file sorgente molto più facile da leggere e da correggere. Alternativamente si dia un nuovo comando `\graphicspath` all'inizio di ciascun capitolo per specificare l'unico percorso per trovare le immagini relative a quel capitolo.

Nello stesso modo la soluzione dell'uso di `\graphicspath` è molto conveniente quando le figure da includere non sono in numero sterminato e possono essere conservate in modo ordinato in una sola cartella.

Un'ultima precisazione; nei sistemi UNIX (Linux, Mac) i link simbolici sono molto usati; nei sistemi Windows essi sono disponibili solo a partire dal sistema operativo Windows Vista; quindi sono una novità abbastanza recente e oggi (2012) non ancora entrati nell'uso "quotidiano" degli utenti Windows.

Un link simbolico è una indicazione che appare in una specifica cartella del sistema operativo, ma che "punta" ad un'altra cartella o ad un file presente in un'altra cartella.

Nei sistemi UNIX si possono creare link simbolici usando il terminale, ponendosi nella cartella dove si vuole creare il link e dando il comando

`ln -s` ⟨*file o cartella puntata completi di percorso*⟩ ⟨*nome del link*⟩

Nei sistemi Windows li si crea in modo analogo, usando la finestra comandi (o prompt dei comandi), mettendosi nella cartella dove si vuole creare il link e dando il comando:

`mklink` ⟨*nome del link*⟩ ⟨*file o cartella puntata completi di percorso*⟩

È opportuno, ma non necessario, che il nome del link abbia le stesse caratteristiche apparenti del file o della cartella puntata; per esempio il solo nome basterà per una cartella, ma per puntare al nome di una immagine in formato `.jpg`, è meglio che il nome del link abbia l'estensione `.jpg`. Se si cancella il link con i soliti comandi, si cancella solo il link, non l'oggetto puntato; per il resto qualunque altra operazione eseguita sul link è come se la si facesse sull'oggetto puntato.

Usando i link si può aggirare la proibizione di usare indirizzi di file che non si trovino nella cartella del file principale o in una delle sue sottocartelle. Si pensi alla immagini contenute in album già organizzati e residenti altrove sul

disco fisso; un link simbolico posto in una cartella subalterna a quella del file principale, può rendere accessibile l'intero album come se lo si fosse copiato nella cartella subalterna.

Però usando i link simbolici l'insieme di cartelle che costituiscono il documento non sono più compatibili con altre macchine, indipendentemente dal sistema operativo, se non hanno esattamente lo stesso contenuto in ogni ramificazione degli alberi di cartelle a partire dalla radice del disco rigido (almeno quelle ramificazioni che coinvolgono il documento). I link simbolici possono essere utili solo per un lavoro "solitario", non in collaborazione con altri coautori, a meno che l'assemblaggio dei vari contributi non sia eseguito da un solo curatore.

10.6.2 Includere le immagini

Per includere le immagini, come si è già detto sopra, bisogna usare il comando `\includegraphics` e bisogna specificargli correttamente la ⟨*lista delle chiavi*⟩.

Questa è costituita da una serie di dichiarazioni separate da virgole e scritte nella forma 'chiave = valore'. Se la chiave rappresenta una affermazione booleana, non è necessario specificarne il valore 'true' ma, quando lo si vuole usare, è necessario specificarne il valore 'false'.

Qui non si esporranno tutte le opzioni disponibili; ci si limiterà a descrivere quelle più comunemente usate, almeno secondo l'esperienza dello scrivente; si rimanda il lettore alla documentazione del pacchetto (`texdoc grfguide`) per avere maggiori dettagli e per le informazioni relative alle altre opzioni.

`bb` Serve per specificare il rettangolo tangente all'immagine con la sintassi:

`bb=` ⟨*llx*⟩ ⟨*lly*⟩ ⟨*urx*⟩ ⟨*ury*⟩

In generale non è necessario specificare le coordinate dei vertici inferiore sinistro e superiore destro del rettangolo tangente; queste coordinate in generale sono contenute nel file grafico da importare. Tuttavia queste informazioni potrebbero mancare oppure potrebbero non corrispondere a quelle del rettangolo tangente. Un caso molto frequente si ha quando con un programma grafico esterno al sistema TEX si crea o si ottiene un disegno o una immagine a mezze tinte da scontornare perché magari il disegno appare su una pagina in formato A4, non su un rettangolino grande quanto il disegno; oppure l'immagine fa parte di una pagina che contiene anche altre parti grafiche che non interessa importare. Le alternative sono quelle di usare la chiave `clip` (o, alternativamente, il comando asteriscato `\includegraphics*`) insieme a una delle chiavi `viewport` o `trim` che verranno descritte poco oltre. La chiave `bb` è particolarmente comoda perché le coordinate da specificare sono facilmente rilevabili mediante il programma `gview` quando si sia specificata l'opportuna opzione di visualizzazione[5] e si

[5] Alcune opzioni sono usabili solo con certi formati, ma le coordinate del cursore sono leggibili sempre, indipendentemente dal formato del file.

sia usato il cursore a croce per individuare i vertici del rettangolo tangente; `gview` funziona sia con i file `.ps` e `.eps`, sia con i file `.pdf`.

`width` serve per specificare la larghezza dell'immagine nel documento finale; generalmente conviene parametrizzare questa dimensione ad un valore legato alla geometria della pagina, invece di specificare un numero di punti, o di millimetri, o di pollici, o di[6]...

Allora è meglio specificare la larghezza nella forma `width=0.5\linewidth` invece di `width=87mm`, perché cambiando layout della pagina la dimensione relativa mantiene le proporzioni, mentre la dimensione assoluta potrebbe dare luogo a inconvenienti non lievi; si pensi per esempio di voler passare da una composizione a piena pagina ad una composizione a due colonne; 87 mm potrebbero andare bene a piena pagina, ma potrebbero essere troppi per una colonna.

Vale la pena di ricordare che `\linewidth` coincide con la giustezza corrente (piena pagina o colonna) salvo che all'interno di certi ambienti potrebbe essere un poco inferiore; è quindi conveniente fare sempre riferimento a `\linewidth` invece che a `\textwidth` o a `\columnwidth`.

Secondo una tradizione conservata dalle norme UNI, i fattori da preferire per moltiplicare la larghezza della linea sono 1, 0,5, 0,2 e le loro radici quadrate. Per le immagini i fattori più piccoli sono da evitare, ovviamente, ma la sequenza di ragione (approssimativa) $\sqrt{2}$ è quella più frequente.[7]

`height` serve per specificare l'altezza dell'immagine riprodotta nel documento; anche in questo caso è meglio fare riferimento all'altezza della pagina, parametrizzando rispetto a `\textheight`; per esempio, conservando uno dei fattori di scala della sequenza UNI, si potrebbe specificare: `height=0.35\textheight`.

`keepaspectratio` Se si sono specificate sia `width` sia `height` la figura potrebbe subire cambiamenti di scala diversi in altezza rispetto alla larghezza; per evitare questo fatto la variabile booleana `keepaspectratio` mantiene inalterato il rapporto di forma dell'immagine da riprodurre e sceglie sia per l'altezza sia per la larghezza il maggiore fra i due rapporti di scala che *non* fa eccedere nessuna delle due dimensioni specificate.

`viewport` serve per specificare la finestra attraverso cui guardare l'immagine; la finestra in effetti fa sì che diventi un nuovo rettangolo tangente all'immagine e, insieme a `clip`, può servire per nascondere le parti dell'immagine che

[6] Le unità di misura dimensionali che il sistema TeX capisce sono i punti (pt), i millimetri (mm), i centimetri (cm), i pollici (in) oltre alle unità legate alle dimensioni del font corrente; l'altezza della 'x' (ex), la larghezza di una 'M' (em). TeX capisce anche altre unità meno frequenti dall'uso un po' particolare, utili specialmente per chi scrive pacchetti di estensione o file di classe.

[7] In sostanza i fattori di scala da preferire per le immagini sono quelli (arrotondati) della sequenza geometrica di ragione $\sqrt{2}$ che contiene il valore unitario: 0,35, 0,5, 0,7, 1, 1,4, 2, 2,8.

non interessano; quindi può servire per scontornare una immagine alla quale non è stato possibile applicare il trattamento specificato nei paragrafi precedenti. Tuttavia può servire anche per mostrare solo un particolare dell'immagine. La sua sintassi è la seguente:

`viewport=` ⟨*llx*⟩ ⟨*lly*⟩ ⟨*urx*⟩ ⟨*ury*⟩

dove ⟨*llx*⟩ ⟨*lly*⟩ sono le coordinate x e y dell'angolo in basso a sinistra della finestra e ⟨*urx*⟩ ⟨*ury*⟩ sono le coordinate x e y dell'angolo in alto a destra della finestra. Queste coordinate sono relative al rettangolo circoscritto originario all'immagine, e questo potrebbe non coincidere con il supporto virtuale sul quale l'immagine digitale è riportata. Non è il caso di preoccuparsi di questo dettaglio perché il 99% delle volte si ha assoluta coincidenza.

`trim` serve ad una funzione simile a quella prodotta da `viewport`, solo che i quattro valori sono le ampiezze delle strisce di immagine da togliere successivamente da sinistra, dal basso, da destra e dall'alto dell'immagine; talvolta è più facile specificare questi valori, che non le coordinate della finestra attraverso cui vedere una parte dell'immagine. La sintassi è la seguente:

`trim=` ⟨*left*⟩ ⟨*bottom*⟩ ⟨*right*⟩ ⟨*top*⟩

Sia con la chiave `viewport` sia con `trim` le dimensioni sono espresse indicando le unità di misura che TEX gestisce; il programma provvede a trasformarle in *punti PostScript*, visto che l'operazione viene svolta in quel linguaggio vero o semplificato; se si desidera esprimere quelle lunghezze direttamente in punti PostScript si usa l'unità di misura `bp`, oppure non si specifica l'unità di misura che per default viene quindi assunta uguale a `bp`.

`clip` serve per dare l'ordine di tagliare effettivamente quanto sporge fuori del rettangolo tangente specificato dalla chiave `bb`, oppure da `trim`, oppure da `viewport`. Se non si esprimesse `clip`, il programma tratterebbe l'immagine come se le sue dimensioni effettive fossero quelle del rettangolo specificato con le chiavi suddette, ma quanto sporgesse fuori da quel rettangolo non verrebbe tagliato via e quindi l'immagine sarebbe ancora tutta integra ma più grande di quanto uno si aspetterebbe. Se invece di `\includegraphics` si usasse la versione asteriscata `\includegraphics*`, allora non sarebbe necessario specificare la parola chiave `clip`, perché il valore `true` con il comando asteriscato è quello di default.

`angle` serve per specificare l'angolo di rotazione (in gradi e in senso antiorario) della figura; quando si usa questa specifica (indicando generalmente $\pm 90°$ o $180°$) insieme a indicazioni di scala del tipo `width=...` bisogna stare

attenti a quale operazione si indica per prima e perciò si esegue per prima; normalmente è conveniente specificare per prima la rotazione, perché è proprio quello che si vuole ottenere.

Bene inteso è possibile specificare anche altre chiavi e i loro valori, ma a quel punto è meglio andare a leggere direttamente la documentazione `grfguide.pdf` che accompagna sempre il sistema TeX e si legge con `texdoc grfguide`.

Nella figura 10.3 sono rappresentati alcuni effetti che si possono ottenere con la stessa fotografia; sotto ogni foto è indicata la lista delle chiavi usata per ottenere il risultato. Nella prima foto in alto a sinistra si ha l'immagine a larghezza piena, fatta alla manifestazione Cheese, patrocinata da Slow Food, che si svolge ogni due anni a Bra (CN); nelle immagini successive si sono applicate diverse chiavi e, in particolare, quando sono state specificate sia `width` sia `height` senza specificare `keepaspectratio`, si è ottenuta una deformazione dell'immagine in senso orizzontale che è decisamente visibile ad occhio nudo. Nelle due immagini dove si è specificato o `trim` o `viewport` insieme alla dichiarazione `clip` l'immagine risulta correttamente ritagliata e ingrandita fino a soddisfare il requisito dell'altezza. Nelle ultime immagini la foto del Minareto di Marrakech "La Koutoubia" è stata ripresa con la fotocamera ruotata di 90° (be', ecco, ehm, circa...) per cui la corretta rappresentazione richiede una rotazione contraria; appare evidente che nella foto di sinistra la base è stata scalata fino ad essere uguale alla larghezza della riga, poi l'immagine è stata ruotata così che la base diventi l'altezza; invece nella foto di destra prima l'immagine è stata ruotata poi la base è stata scalata alla larghezza della riga e solo in questo modo si è ottenuto il risultato desiderato.

È bene notare che le operazioni eseguite con le chiavi `trim` e `viewport` non sembrano così semplici da fare perché non sono generalmente note le dimensioni naturali dell'immagine; tuttavia è bene ricordare che le dimensioni del rettangolo tangente oppure del rettangolo che deve funzionare da finestra sono sempre rilevabili con il cursore a croce del programma gview, che si assume sia sempre disponibile al compositore.

In mancanza di tale programma, vale la pena di ricordare che ogni volta che si lancia LaTeX o pdfLaTeX, viene prodotto un file con estensione `.log` e con il nome uguale a quello del file sorgente che si è appena compilato. In questo file, un normale file testuale contenente solo caratteri ASCII, quindi leggibile senza problemi con qualunque programma di elaborazione testi, quando viene importato per la prima volta un dato file grafico contenente un'immagine, ne viene scritto il tipo e ne vengono fornite le dimensioni 'base per altezza' della 'bounding box' in punti tipografici; con queste dimensioni è possibile fare le debite proporzioni di ciò che si desidera ritagliare via e quindi non è difficile trovare i valori numerici da usare; ricordiamoci infatti che le coordinate da specificare sono quelle del file di partenza, non del file riprodotto, quindi è necessario riferirsi alle coordinate del file di partenza.

`width=\linewidth`

`height=0.5\linewidth`

`width=\linewidth,`
`height=0.5\linewidth`

`width=\linewidth,`
`height=0.5\linewidth,`
`keepaspectratio`

`height=0.5\linewidth,`
`trim=100 50 150 120, clip`

`height=0.5\linewidth,`
`viewport=100 50 492 362, clip`

`width=\linewidth, angle=90`

`angle=90, width=\linewidth`

Figura 10.3: Due foto trattate con diverse chiavi

Capitolo 11

La bibliografia

11.1 Introduzione

Come si è già detto, LaTeX compone la bibliografia come un testo speciale costituito da un elenco di voci etichettate con un'etichetta di vari stili, ma citabili con una chiave mnemonica scelta dall'autore o fissata in altro modo.

Il problema della bibliografia non è tanto quello di preparare l'elenco con la sintassi descritta per l'ambiente *thebibliography*, quanto quello di elencare i riferimenti in un ordine logico, nello scrivere tutte le informazioni necessarie al reperimento dell'opera citata, e, specialmente, nell'essere coerenti nello stile di presentazione di tutte queste informazioni.

Vale la pena di ricordare che le bibliografie e gli elenchi bibliografici sono regolati dalla norma ISO 690, [28], e dalla norma UNI 6017, [61]. Queste norme specificano quali siano le informazioni necessarie e quelle facoltative richieste da un riferimento correttamente conforme alle norme. In buona sostanza sono tutte le informazioni che permettono di identificare univocamente il riferimento citato. Le norme non specificano in che stile esse debbano essere stampate; l'importante è che informazioni dello stesso tipo siano stampate con lo stesso stile. I programmi BibTeX e biberconsentono di rispettare le norme e di mantenere lo stile di composizione in modo uniforme per tutti i vari riferimenti di vario tipo. Forse merita ricordare che le opere antiche o classiche vanno comunque identificate univocamente, perché le varie edizioni possono essere differenti, più o meno complete, aggiornate sui manoscritti o codici di più recente disponibilità, eccetera. Insomma non basta citare "Manzoni, *I promessi sposi*" senza citare la particolare versione, edizione critica, edizione commentata a cui ci si sta riferendo.

11.2 I programmi di estrazione bibliografica

I programmi BibTeX e biber servono per rispondere alle necessità descritte sopra in modo tipograficamente perfetto e versatile. Essi servono in sostanza per estrarre le informazioni necessarie da un database bibliografico e per comporre un file adatto a LaTeX per comporre la bibliografia nel documento finale.

In questo paragrafo ci riferiremo specialmente a BibTeX, ma molto di quello che si dirà qui sarà valido anche per biber del quale si dirà di più in un paragrafo successivo quando si esporrà il pacchetto *biblatex*.

Questi programmi richiedono che si disponga di un database di riferimenti bibliografici scritto in una certa maniera che fra poco verrà sommariamente illustrata; essi traggono da questo database solo i riferimenti citati in un dato documento e ne preparano l'elenco, generalmente ordinato alfabeticamente secondo il cognome del primo autore, nella forma corretta per essere trattato da LaTeX e composto secondo quanto desiderato.

11.2.1 Come specificare lo stile bibliografico

Solo BibTeX. Nel preambolo del documento bisogna specificare lo stile compositivo dell'elenco bibliografico; lo si fa con il comando `\bibliographystyle` secondo la seguente grammatica:

`\bibliographystyle{`⟨*stile bibliografico*⟩`}`

Eventualmente questo comando può essere preceduto dalla specificazione di un pacchetto che interagisce con la composizione della bibliografia mettendo a disposizione diverse varianti dei comandi di citazione, per esempio il pacchetto *natbib*.

11.2.2 Come comporre la bibliografia

Solo BibTeX. Si inserisce il comando `\bibliography` con la seguente sintassi nel punto dove si vuole comporre la bibliografia

`\bibliography{`⟨*database bibliografico*⟩`}`

dove ⟨*database bibliografico*⟩ è il nome del database bibliografico che si intende usare; disponendo di più database e dovendo citare opere elencate in diversi database, il ⟨*database bibliografico*⟩ può consistere in una lista di nomi separati da virgole.

Gli stili bibliografici distribuiti di default sono file con l'estensione `.bst` e si trovano nella cartella `.../bibtex/bst/` dove sono raccolti in diverse sottocartelle. Gli stili più diffusi sono `unsrt.bst` e `plain.bst`; il primo non esegue nessun ordinamento, ma le opere sono elencate nello stesso ordine in cui vengono citate; con il secondo stile le opere vengono ordinate secondo l'ordine alfabetico del primo o unico autore, o quanto possa sembrare il nome di un autore al programma,

poi per anno e poi per ordine alfabetico del titolo. Con questi stili le opere vengono citate per numero d'ordine racchiuso fra parentesi quadre, come si fa abitualmente nei lavori tecnico–scientifici. Per citazioni diverse bisogna affidarsi al pacchetto di estensione *natbib* alla cui documentazione si rinvia per i dettagli necessari.

Tutti questi stili ipotizzano che la bibliografia sia scritta in inglese; non è difficile crearsi degli stili personalizzati, in particolare in italiano, ricorrendo al file `makebst.tex` che provvede a tutto il necessario pur di rispondere 'correttamente' ad un certo numero di domande; questo file va elaborato con LaTeX e richiede che si modifichi un file modello dal nome `merlin.mbs`; tutta l'operazione non è difficile, ma richiede che si sia già acquisita una certa dimestichezza con i database bibliografici e con i vari stili bibliografici.

Naturalmente per le varie piattaforme esistono delle interfacce grafiche che consentono di comporre i database bibliografici nel modo corretto; tanto per citarne uno, per il Mac OS X esiste (ed è distribuito mediante MacTeX insieme al sistema TeX) il programma BibDesk.

Per tutte le piattaforme esiste il programma in linguaggio Java JabRef che fa più o meno le stesse cose.

Inoltre diversi shell editor, come per esempio TeXstudio, consentono di creare direttamente il database bibliografico.

11.2.3 Chiavi e citazioni

Ma la cosa non finisce qui; per avere la bibliografia composta, il programma deve disporre delle chiavi di citazione; perciò bisogna lanciare LaTeX; appena ha finito e non ci sono errori, tranne eventualmente avvisi che alcune chiavi di citazione non sono state 'risolte' (il che significa che LaTeX non ha trovato altro che le chiavi, cioè gli argomenti dei comandi `\cite`, ma non sa a che cosa corrispondano queste chiavi), bisogna passare alla fase successiva.

In certe circostanze sarebbe desiderabile introdurre nell'elenco bibliografico anche opere non espressamente citate nel testo, al limite tutte le opere contenute nel database bibliografico. Per ottenere questo scopo all'interno del documento (cioè *non* nel preambolo) si può usare il comando `\nocite` con la seguente sintassi:

`\nocite{`⟨*elenco delle chiavi*⟩`}`
oppure
`\nocite{*}`

dove ⟨*elenco delle chiavi*⟩ è una lista di una o più chiavi di citazione di opere che si vogliono elencare nell'elenco bibliografico anche se non sono state usate come argomenti del comando `\cite` all'interno del documento. Se invece dell'elenco delle chiavi si usa l'asterisco, allora nell'elenco bibliografico compariranno tutte le opere contenute nel database bibliografico; esistono delle circostanze in cui vale la pena di comporre un database apposta (traendo le informazioni da altri database mediante la nota tecnica del "taglia e incolla") in modo da creare

nel documento composto una bibliografia di consultazione, non di riferimento come solitamente avviene quando le opere elencate vengono citate nel corpo del documento.

A questo punto bisogna lanciare BibTeX o da una finestra comandi o terminal o console, oppure bisogna cliccare sull'apposito bottone dello *shell editor*. In questo modo BibTeX apre il o i database specificati, cerca le voci corrispondenti alle opere citate in base alle chiavi raccolte da LaTeX nel giro precedente; le ordina secondo i criteri di ordinamento specificati nello stile bibliografico e finalmente genera un file con estensione `.bbl` che contiene in linguaggio LaTeX tutto quello che occorre per comporre la bibliografia.

Finalmente si lancia nuovamente LaTeX e questa volta viene composta la bibliografia e vengono decifrate le chiavi; ma non è finita; le chiavi sono decifrate, ma non sono ancora state usate per le citazioni; bisogna lanciare ancora una volta LaTeX in modo che possa usare le chiavi decifrate al giro precedente così che le informazioni ci siano tutte e siano state usate tutte al posto giusto. Una ulteriore passata del documento a LaTeX non guasta; serve per giustificare correttamente il documento e può rendere corretti i riferimenti incrociati che avrebbero potuto risultare modificati dall'uso delle chiavi.

Esercizio 11.1 Perché dopo la seconda volta che si lancia LaTeX dopo aver lanciato BibTeX i riferimenti incrociati potrebbero non essere corretti? Perché quindi è necessario lanciare LaTeX ancora una volta?

In sostanza la generazione e composizione della bibliografia avviene in quattro o cinque fasi così:

(1) LaTeX (2) BibTeX (3) LaTeX (4) LaTeX (5) LaTeX

Non è il caso di spaventarsi per questa (apparente) complicazione; bisogna ricordare prima di tutto che un buono *shell editor* dispone quasi sempre di un bottone da cliccare che esegue tutte le passate che occorrono per avere la bibliografia composta correttamente e i riferimenti incrociati fra loro perfettamente coerenti; a cosa servirebbe sennò un buono *shell editor*?

Nello stesso tempo, se lo *shell editor* non disponesse di quel magico bottone e fosse necessario procedere a mano attraverso la finestra comandi o il terminal o la console, eseguire quei cinque comandi richiede una manciata di secondi; la compilazione di un testo di molte centinaia di pagine, diciamo 700, richiede con i PC o laptop moderni qualche decina di secondi, BibTeX richiede qualche secondo; l'intera operazione da eseguirsi una volta sola alla fine della lavorazione del documento può richiedere un paio di minuti; siccome anche l'indice analitico richiede lo stesso numero di passate, si può ottimizzare l'operazione prendendo due piccioni con una fava. Non è terribile.

11.3 I database bibliografici

I database bibliografici sono dei file scritti in testo puro con i caratteri ASCII o anche con i caratteri accentati gestibili con l'input encoding preferito; per la trasportabilità sarebbe meglio che si trattasse di caratteri ASCII puri (codici a 7 bit) e quindi i segni letterali con diacritici dovrebbero essere rappresentati con le sequenze per gli accenti che non sono state descritte finora da nessuna parte, ma si rinvia a qualunque guida o manuale 'oldstyle' di LaTeX[1]; qui non si vuole fare gli schizzinosi circa le tradizionali modalità di scrivere le lettere accentate immaginando di usare una tastiera USA, priva di tasti speciali per i caratteri accentati; semplicemente si ritiene molto più comodo servirsi delle potenzialità del proprio PC o laptop per gestire i caratteri con diacritici e le potenzialità dei file di estensione di LaTeX per comporre correttamente quelle lettere speciali.

Il punto è che in Europa, e in Italia in particolare, è difficile comprare un PC o un laptop con una tastiera diversa da quella nazionale. In Italia, in particolare, scrivere le vecchie sequenze LaTeX per gli accenti sarebbe particolarmente fastidioso perché l'accento italiano più diffuso è quello grave e dalla tastiera italiana manca qualunque tasto o semplice combinazione di tasti che consenta di introdurlo; bisogna ricorrere al tastierino numerico, il che con i laptop si può fare ma richiede ulteriori tasti da premere.

Dopo questa lunga premessa del fatto che sarebbe meglio usare i caratteri ASCII, ma che con la tastiera italiana la cosa è particolarmente fastidiosa, veniamo al database; ogni voce del database comincia con la dichiarazione del tipo di documento a cui la voce si riferisce; questa dichiarazione è una delle seguenti parole: `article`, `book`, `booklet`, `conference`, `inbook`, `incollection`, `inproceedings`, `manual`, `masterthesis`, `misc`, `phdthesis`, `proceedings`, `techreport`, `unpublished`. Ognuna di queste categorie di documenti è spiegata nella documentazione di BibTeX, in `.../doc/bibtex/btxdoc.pdf`.

In base alle norme ISO ognuna di queste tipologie di documento richiede certe informazioni, quindi nel descrivere il documento bisogna usare un certo numero di parole chiave; se per un certo tipo di riferimento una voce obbligatoria manca, quando BibTeX viene eseguito vengono emessi degli avvertimenti del fatto che la citazione tale è priva della specificazione talaltra. Le chiavi facoltative e obbligatorie per ogni tipologia sono specificate nella documentazione citata.

Qui ci limiteremo a qualche piccolo esempio commentando ciò che appare nelle voci del database.

```
@manual{man:refbib,
  title=       {Documentation - {B}ibliographical references -
               {E}ssential and supplementary  elements},
  note=        {Norma {ISO} 690--1975},
  organization={International Organization for Standardization},
  address=     {Ginevra},
  year=        {1975},
```

[1] Si veda comunque il paragrafo 29.2.5 e la tabella 29.1.

```
  key=          {ISO 690}
}

@manual{man:patashnik,
  author=       {Oren Patashnik},
  title=        {\textsc{Bib}{\TeX}ing},
  organization=}{TUG},
  year=         {1988},
  note=         {Il file \texttt{btxdoc.dvi} contenente questa
                documentazione generalmente è già presente nella
                cartella \texttt{/texmf\slash doc\slash bibtex/}
                poiché fa parte integrante del sistema \TeX}
}

@book{book:latexe,
  author=       {Leslie Lamport},
  title=        {\LaTeX, a document preparation system ---
                User's guide and reference manual},
  publisher=    {Addison--Wesley Publ. Co.},
  address=      {Reading, Mass.},
  year=         {1994},
  edition=      {2}
}
```

Come si vede dai tre esempi di voci bibliografiche riportati sopra, la tipologia del documento viene sempre preceduta dal segno `@` e il contenuto della voce è racchiuso fra parentesi graffe. Il primo elemento contenuto fra queste parentesi è la chiave di citazione; nel file LaTeX, quindi, per esempio, le norme ISO per le citazioni bibliografiche *devono* essere citate con `\cite{man:refbib}` e non con un'altra chiave che piaccia di più.

Le altre informazioni relative al documento citato sono specificate mediante una espressione del tipo:

⟨*parola-chiave*⟩ `={`⟨*testo*⟩`},`

dove l'ultima virgola può essere omessa solo nell'espressione finale. Ciò che nel ⟨*testo*⟩ compare fra graffe non subisce modificazioni quando viene trascritto nel file di uscita; altrimenti BibTeX spesso cambia le maiuscole in minuscole con il risultato di alterare quello che si vuole scrivere, oppure di cambiare l'ortografia di macro che verranno poi interpretate da LaTeX.

Le norme ISO sono classificate come `manual`; questa tipologia non ha bisogno dell'indicazione dell'autore; per consentire un qualunque ordinamento, allora, è opportuno inserire una parola chiave `key` che serve come chiave di ordinamento in mancanza di altre chiavi predefinite per questo scopo.

Anche la documentazione di BibTeX è classificata come manuale, ma l'autore è noto ed indicato, per cui non è necessaria una chiave supplementare.

L'ordine in cui vengono scritte le ⟨*parole-chiave*⟩ e le relative espressioni non è essenziale; l'importate è che ci siano tutte quelle relative ad una data tipologia ed eventualmente ci siano quelle facoltative che possono agevolare l'ordinamento secondo qualche criterio dipendente dallo stile bibliografico prescelto.

Dipende dal compositore decidere se conviene predisporsi e tenere aggiornato uno o più database bibliografici oppure se comporre di volta in volta la lista dei riferimenti mediante la compilazione del contenuto dell'ambiente *thebibliography*; inizialmente si è un po' restii a creare un database bibliografico; poi se si supera questo scoglio iniziale se ne apprezza completamente la validità e la comodità.

Qui vorremmo consigliare di fare uso dei database bibliografici; tutto sommato bisogna inizialmente superare uno scoglio di tipo psicologico, ma via via che il proprio database (o i propri database distinti per tipologia di contenuto) si arricchiscono di voci, il lavoro per tenerli aggiornati diventa sempre più modesto. In compenso l'uso degli stili bibliografici consente di comporre le informazioni richieste in modo coerente; se la casa editrice a cui si manda un lavoro vuole una bibliografia composta in un altro modo, il database rimane lo stesso, ma la nuova composizione viene eseguita in sostanza lanciando BibTeX e lasciando fare tutto il resto a LaTeX. Si immagini di dover invece correggere a mano una lista composta direttamente entro l'ambiente *thebibiography*; se la bibliografia consiste anche soltanto di poche decine di voci, le possibilità di commettere errori, ma soprattutto, di non essere coerenti sono numerosissime, e il risultato finale non sarà all'altezza del tempo impiegato.

Tipicamente gli elenchi formati secondo determinati stili sono operazioni da eseguire a macchina; la bibliografia è uno di questi elenchi; la conclusione del sillogismo è ovvia: si componga la bibliografia a macchina.

Piuttosto, se gli stili disponibili non sono adeguati e se fosse necessario compilare degli stili nuovi, non si trascuri la possibilità di servirsi del file `makebst.tex`; all'inizio, poiché le domande a cui bisogna rispondere sono in inglese, ci si può trovare un po' frastornati dai termini tecnici insoliti; la prima volta che si compie questa operazione è normale commettere errori e dover eseguire di nuovo la creazione dello stile, ma il gioco vale la candela e il risultato finale di solito è di grande soddisfazione.

11.4 Quanti database bibliografici?

È chiaro che questa è una domanda insolita e alla quale è difficile dare una risposta. Già prima si è accennato alla possibilità di creare database bibliografici divisi per argomenti; questa è già una risposta parziale, ma bisogna associarla al tipo di documento che si vuole comporre.

In un articolo o rapporto scientifico è possibile che si debbano riportare solo le voci bibliografici dei riferimenti citati; perciò anche un unico database generale può essere utile, come potrebbero esserlo diversi database specialistici. Invece un testo di tipo illustrativo (in certe università si usa l'orrenda parola *compilativo*, a cui è spesso associata una connotazione negativa – se il testo

raccoglie solo titoli di articoli e di riviste, effettivamente serve a poco, ma se si tratta di una raccolta ragionata, commentata, verificata, di opere che descrivono davvero *lo stato dell'arte* allora il testo diventa tutt'altro che spregevole, anzi è una benedizione!) allora è probabile che l'elenco contenga anche opere non citate nel testo, ma venga usato il comando `\nocite{*}` per comporre nella bibliografia l'intero database bibliografico.

Come gestire e mantenere diversi database bibliografici? Certamente usando programmi adatti, ma sarebbe meglio evitare sovrapposizioni; sarebbe meglio che ogni riferimento compaia una sola volta nei database generali. Come fare? Usando i link simbolici, oggi disponibili anche sulle macchine Windows a partire da Windows Vista.

L'operazione si svolge in questo modo: in una cartella personale si raccolgono tutti i database bibliografici che interessano, magari divisi per argomenti o con qualche altro criterio Si curerà di controllare che ogni opera compaia una sola volta e non venga ripetuta in opere diverse; si controllerà che le chiavi siano davvero distinte. A questo proposito conviene moltissimo comporre le chiavi mediante un prefisso, tipo 'libro', 'manuale', 'articolo', eccetera (meglio se abbreviati in modo univoco), separato dal nome 'proprio' dell'opera citata mediante, per esempio due punti, come si è indicato nell'esempio della pagina 235. Il nome proprio può essere formato dal cognome del primo autore seguito da altre informazioni; si può usare qualunque carattere tranne la virgola, ma è meglio evitare gli spazi. Sono cose già dette, ma conviene ripeterle.

Via via che si compone un particolare documento si tengono aperti anche i file dei database bibliografici al fine di riportare le chiavi di chiamata semplicemente copiandole da ciascun database (senza perdere l'occasione di verificare che gli elementi necessari all'identificazione dell'opera siano presenti, non debbano essere aggiornati, e cose di questo genere); se per questa operazione bisogna aprire un altro database, si coglie l'occasione per trascriverne il nome nell'elenco di database passati al comando `\bibliography`. Nel documento *non* si usa il comando `\nocite{*}`, ma si lascia a BibTeX il compito di estrarre dai database tutti i riferimenti citati e di comporli come si deve.

Ma BibTeX cerca i database nella stessa cartella dove si trova il file principale del documento che si sta componendo, mentre i database si trovano in una cartella personale dell'utente dedicata a conservare tutti i database bibliografici. Allora nella cartella che contiene il file principale si aggiungono i link simbolici ai file che si trovano altrove. Per le macchine di tipo UNIX (Linux, Mac, eccetera) si userà la sintassi:

`ln -s` ⟨*percorso completo del file vero*⟩ ⟨*nome del link simbolico*⟩

mentre su una macchina Windows con sistema operativo Vista o successivo, il comando diventa:

`mklink` ⟨*nome del link simbolico*⟩ ⟨*percorso completo del file vero*⟩

Solitamente si usa come ⟨*nome del link simbolico*⟩ il nome del *file vero* privato di tutto il percorso che vi conduce, di cartella in cartella, a partire dal nome

del disco fisso. Il link simbolico fa credere al sistema operativo che nella cartella dove c'è il nome del link simbolico ci sia davvero il file originale. In questo modo l'aggiornamento o la correzione del *file vero* esistente in un'unica copia, si riflette automaticamente su tutti i documenti che hanno fatto riferimento a quel database tramite il link simbolico.

Diversa è la situazione se si vuole comporre un documento informativo dove si vogliono riportare tutti i riferimenti di un database bibliografico, anche quelli non citati. Per questo scopo è necessario creare nella cartella del file principale del documento un nuovo database bibliografico che contenga tutte e sole le voci che si desidera riportare nella bibliografia. Questa non è una operazione difficile da fare, basta copiare dai database principali i dati riferiti solo alle opere che interessando, incollandone le righe nel nuovo database dedicato al particolare documento. Dopo `\nocite{*}` e BibTeX fanno il resto, componendo la bibliografia come si deve.

È incredibile quanto tempo si risparmia a seguire le indicazioni riportate sopra. Ma è anche incredibile quanto risultino composte bene le bibliografie con questi strumenti; s'intende che i database bibliografici non devono contenere errori – questo va da sé!

11.5 Il pacchetto *biblatex*

Oggi, poi, usando il pacchetto di estensione *biblatex*, si possono ottenere bibliografie perfette, composte anche con stili diversi da quelli standard disponibili nella distribuzione del sistema TeX.

Il pacchetto *biblatex* funziona come un file di stile bibliografico, ma provvede alla definizione di una quantità di comandi interni che consentono di comporre la bibliografia in modi molto più complessi di quelli che si possono ottenere con BibTeX e i file `.bst`. Le variazioni si specificano mediante una lista di opzioni espresse nella forma *chiave* = *valore*; esse sono numerosissime e bisogna riferirsi al manuale di *biblatex* per sapere quali usare e come usarle.

Tipicamente un documento che faccia uso di collegamenti ipertestuali e il cui database bibliografico si chiami `Bibliografia.bib` viene composto inserendo nel preambolo una specificazione del tipo

```
\usepackage[autostyle]{csquotes}
\usepackage[style=philosophy-modern,hyperref,natbib,
   backref,square,backend=biber]{biblatex}
\addbibresource{Bibliografia.bib}
```

e poi scrivendo verso la fine del documento, nel punto in cui si desidera comporre la bibliografia, il comando

```
\printbibliography
```

Il pacchetto *csquotes* serve per usare le virgolette uncinate (caporali) nelle citazioni; lo stile di composizione specificato, mediante l'opzione *style*=`philosophy-modern`,

è uno stile con certe caratteristiche usate specialmente in saggi di tipo umanistico; l'opzione *hyperref* serve per comporre correttamente i collegamenti ipertestuali, ma richiede che il pacchetto `hyperref` venga poi caricato; l'opzione *natbib* serve per usare certi stili di citazione come quelli definiti dal pacchetto `natbib`; l'opzione *backref* serve per creare i collegamenti ipertestuali a ritroso; l'opzione *square* serve per usare le parentesi quadre per le citazioni; infine l'opzione *backend*=`biber` specifica che bisogna usare biber come estrattore bibliografico; si fosse specificato BibTeX, si sareebbe usato il vecchio estrattore, ma con funzionalità ridotte rispetto a biber.

Dal 2012 l'estrattore bibliografico da usare con `biblatex` è biber che dovrebbe essere più versatile del programma BibTeX. Il vantaggio principale è quello che può gestire molto meglio i caratteri codificati in UNICODE, cosa che diventa importantissima quando si debbano comporre bibliografie che contengano dei riferimenti in lingue scritte con alfabeti diversi da quello latino. Il pacchetto `biblatex` a sua volta può funzionare sia con il tradizionale BibTeX sia con il più recente progbiber; la sua documentazione marca in modo chiaro quali prestazioni non siano ottenibili con BibTeX; tuttavia è possibile specificare fra le opzioni nella invocazione del pacchetto biblatex quale dei due programmi di composizione della bibliografia si desidera usare, quindi volendo si può continuare ad usare il vecchio estrattore bibliografico. Si consiglia comunque di usare biber e di configurare lo *shell editor* affinché esso usi biber quando gli si chiede di comporre una bibliografia. Invece di riconfigurare lo shelle editor, si può anche usare la riga magica

```
% !BIB TS-program = biber
```

se lo shell editor è in grado di interpretare le righe magiche.

A parte queste differenze, sostanziali, ma specifiche di `biblatex`, il database bibliografico da usare con questo pacchetto, il file con estensione `.bib`, è della stessa forma di quello che si usa con BibTeX; volendo si potrebbero usare anche altre parole chiave in questo data base bibliografico e per i dettagli si può consultare la documentazione di `biblatex`.

Capitolo 12

LaTeX: indici e glossari

12.1 Introduzione

Gli indici analitici e i glossari possono essere molto utili nelle opere di consultazione. LaTeX offre un grande aiuto per la compilazione di questi elenchi, ma la parte difficile resta quella dell'autore che deve decidere che cosa inviare all'indice analitico o al glossario, se deve usare voci indipendenti o subordinate, se deve distinguere il modo di scrivere le pagine a seconda di come il lemma di quella voce viene trattato; definizione, esempio d'uso, applicazione, eccetera.

12.2 L'indice analitico

L'indice analitico è un elenco di voci ordinate alfabeticamente anche con alcuni livelli di subordinazione, e vicino ad ogni lemma viene inserito il numero della pagina dove quel lemma viene trattato. Il lemma può essere trattato in diversi punti del testo, quindi di fianco al lemma può comparire un elenco di numeri di pagina.

La raccolta di queste informazioni viene fatta mediante il comando `\index`; la sintassi è

```
\index{⟨lemma⟩}
```

Se nel preambolo c'è l'istruzione `\makeindex`, l'informazione specifica relativa al ⟨*lemma*⟩ viene inviata ad un file ausiliario da elaborare in un secondo tempo; se quell'istruzione manca, nel file ausiliario non viene scritto niente. Quando i calcolatori erano lenti, era importante non rallentare l'elaborazione del file sorgente con operazioni non strettamente necessarie; oggi la cosa non è più così importante, ma visto che l'indice analitico è l'ultima cosa che si compone in un libro, è bene non perdere tempo con questa informazione finché non si è alla fine o quasi alla fine.

L'informazione che viene scritta da LaTeX nel file ausiliario (il cui nome ha l'estensione `.idx`) è del tipo:

`\indexentry{`⟨*lemma*⟩`}{`⟨*pagina*⟩`}`

Per ottenere questa raccolta di dati basta collocare l'istruzione `\index` con il suo argomento 'attaccata', cioè senza lasciare spazi, alla parola che si vuole indicizzare. Per esempio:

```
... la tecnologia dei transistori\index{transistore}
      NPN\index{transistore!NPN} ...
```

12.2.1 Il programma makeindex

Per elaborare il file `.idx` in modo da metterlo nella forma giusta, cioè in ordine alfabetico, strutturato per lemmi, sottolemmi e sotto-sottolemmi, con gli elenchi di numeri di pagine senza doppioni o con intervalli, eventualmente composti con caratteri diversi a seconda dell'uso del lemma, esiste il programma makeindex; si tratta di un programma da lanciare a mano dalla finestra comandi o dal terminal o dalla console oppure premendo l'opportuno pulsante dello *shell editor*.

Questo programma esamina il ⟨*lemma*⟩ come trascritto senza modifiche dall'argomento di `\index` al file `.idx` e lo elabora secondo quanto vi trova indicato.

Infatti quello che appare in ⟨*lemma*⟩ non è necessariamente il solo 'lemma' ma anche informazioni ausiliarie che facilitano il compito del programma makeindex.

Nel primo esempio elementare indicato sopra, il ⟨*lemma*⟩ è costituito semplicemente dalla parola 'transistore'; questa parola serve a diversi scopi:

1. indica che si tratta di un lemma di primo livello;
2. indica che la chiave con cui eseguire l'ordinamento alfabetico è 'transistore';
3. indica che va riportato nell'indice analitico tale e quale, vale a dire stampato con il font di default con cui viene composto l'indice;
4. indica che il numero della pagina in cui compare va scritto anch'esso con il font di default.

Strutturando il ⟨*lemma*⟩ in modo più articolato le varie funzioni indicate nella precedente enumerazione possono essere trattate in modo diverso così da far apparire il livello del 'lemma' nell'indice, per indicare una diversa chiave di ordinamento, per indicare come scrivere il 'lemma' nell'indice e come stampare il numero della pagina; la strutturazione consente anche di indicare l'inizio e la fine degli intervalli di pagine, per inserire nell'indice il rinvio ad un altro lemma, eccetera.

I dettagli sono spiegati bene nella documentazione del programma makeindex che si trova nella cartella `.../doc/makeindex/` dove non solo è contenuto il file `makeindex.dvi` ma anche un semplice tutorial `ind.dvi`.

Sommariamente qui si riportano due esempi:

C. Beccari – picture

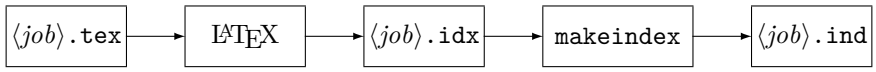


Figura 12.1: Le fasi per la produzione dell'indice analitico

1. Scrivendo

   ```
   \index{transistore!NPN}
   ```

   il lemma 'NPN' viene scritto strutturato sotto il lemma 'transistore'.

2. Scrivendo

   ```
   \index{ambiente!enumerate@\textsf{enumerate}}
   ```

   il lemma 'enumerate' viene scritto con il font senza grazie e viene strutturato sotto al lemma 'ambiente'.

Naturalmente per sfruttare tutta la forza del programma makeindex bisogna leggere attentamente la documentazione citata e prendere esempio dal tutorial menzionato sopra.

12.2.2 La composizione effettiva dell'indice analitico

La composizione effettiva dell'indice analitico avviene specificando nel file sorgente per LaTeX:

```
\input{⟨file⟩.ind}
```

dove ⟨*file*⟩ è il nome del 'job' e `.ind` ne è l'estensione; Il 'job' coincide con il nome del file sorgente per LaTeX che ha dato luogo alla raccolta dei dati per l'esecuzione dell'indice analitico ed è stato elaborato da `makeindex` secondo lo schema della figura 12.1. Questo file contiene anche i comandi di apertura e di chiusura dell'ambiente *theindex*, quindi se non si vuole altro, basta semplicemente eseguire l'inclusione di questo file tramite il comando `\input`.

Come la figura dimostra chiaramente, per ottenere la composizione definitiva dell'indice analitico bisogna lanciare LaTeX ancora una volta, perché finché il file ⟨*job*⟩`.ind` non è disponibile, l'indice non può essere composto (vedi però più avanti).

Usando il pacchetto di estensione *makeidx* è possibile servirsi del semplice comando `\printindex` che provvede da solo a generare l'ambiente e a comporre l'indice analitico.

Siccome i numeri delle pagine devono essere quelli definitivi, è evidente che la composizione dell'indice analitico deve essere in assoluto l'ultima operazione da fare; se si deve eseguire una qualche correzione nel testo, questa correzione comporta anche la correzione dell'indice analitico, vale a dire la triplice esecuzione

di LaTeX con l'esecuzione di makeindex dopo la prima di LaTeX. Per fortuna gli *shell editor* hanno generalmente un bottone solo che esegue tutta questa procedura, necessaria anche per comporre la bibliografia usando BibTeX o biber.

12.3 Il glossario

Un glossario è un elenco di voci accompagnate da una breve spiegazione; talvolta appare anche la pagina del testo dove viene riportata una spiegazione meno succinta.

Il procedimento per produrre un glossario è simile a quello usato per comporre l'indice analitico, ma è un po' più artigianale, come si vedrà fra poco.

L'inizio consiste nel marcare il testo sorgente in una maniera simile a quella dell'indice analitico:

```
\glossary{⟨voce⟩}
```

Lanciando LaTeX, se il preambolo contiene l'istruzione `\makeglossary`, viene prodotto anche un file ⟨*job*⟩`.glo` che contiene tutte le voci di cui si vuol generare il glossario accompagnate dal numero della pagina dove esse compaiono insieme ad una loro spiegazione non succinta.

Lanciando il programma makeindex dalla linea di comando

```
makeindex -s gglo.ist -o ⟨job⟩.gls ⟨job⟩.glo
```

si produce l'elenco alfabetizzato ⟨*job*⟩`.gls` delle parole di cui si desidera comporre il glossario;

Bisogna aprire questo file e aggiungere la breve spiegazione oltre che a correggere la forma di stampa del numero di pagina (per esempio per cancellarlo, se esso non svolge nessuna funzione specifica, come quando il glossario non è altro che un elenco di simboli o di abbreviazioni); poi bisogna comporlo in una maniera simile a quello che si fa per la composizione dell'indice analitico.

In questo senso la composizione del glossario è un po' più artigianale, in quanto bisogna intervenire ancora sul file alfabetizzato.

Esistono diversi pacchetti in rete per comporre glossari o simili elenchi commentati. Qui si suggerisce il pacchetto *glossaries* già presente in ogni installazione di TeX Live completa; esso offre una maniera comoda per scrivere la ⟨*voce*⟩ in modo che contenga direttamente anche la descrizione della voce, così che quando si compone il file alfabetizzato esso sia già pronto per essere composto con LaTeX. Il pacchetto, ovviamente, definisce il comando `\printglossary`, del tutto simile nelle sue funzioni a `\printindex`.

Si raccomanda la lettura attenta della documentazione del pacchetto *glossaries* perché esso offre diverse altre funzioni avanzate che possono tornare molto comode per la generazione di un glossario e per l'amministrazione delle voci da inserirvi.

Il manuale del pacchetto si legge con il solito comando da terminale o prompt dei comandi `texdoc glossaries`; siccome il pacchetto svolge moltissime funzioni,

il manuale è piuttosto esteso; ma l'autrice Nicola Talbot ha scritto un'altra guida introduttiva, specialmente adatta ai nuovi utenti; la si legge con il comando `texdoc glossariesbegin`. Siccome *glossaries* rimpiazza il precedente e ormai obsoleto pacchetto *glossary*, l'autrice ha anche predisposto una terza guida per migrare precedenti documenti al nuovo pacchetto;la si legge con `texdoc glossary2glossaries`. È quanto mai opportuno riferirsi a queste guide, perché il pacchetto è molto versatile e senza queste guide non si può usare efficientemente questo pacchetto.

12.4 Come modificare la composizione dell'indice analitico

Quanto viene esposto qui vale anche per modificare la composizione del glossario.

Per entrambi il file che compone l'elenco richiesto è prodotto dall'esecuzione del programma makeindex; per l'indice analitico viene prodotto il file ⟨*job*⟩`.ind` mentre per il glossario viene prodotto il file ⟨*job*⟩`.gls`. Entrambi questi file racchiudono l'elenco da comporre dentro due specifici ambienti *theindex* e *theglossary*. Come ogni voce sia articolata dipende da come si siano usati i comandi `\index` o `\glossary`, e quindi da come le varie opzioni siano state elaborate da makeindex per produrre i file di uscita. Cambiando opzioni, chiaramente, cambia anche la composizione.

Tuttavia qui si vuole richiamare l'attenzione su un altro punto e per esemplificare ci riferiremo all'indice analitico; per il glossario valgono considerazioni del tutto analoghe.

La formattazione dell'indice analitico dipende da come è definito l'ambiente *theindex* oltre che dal contenuto del file ⟨*job*⟩`.ind`. Cambiare formattazione dell'indice analitico, implica la capacità di cambiare la definizione dell'ambiente *theindex*. Si tratta di modificare la definizione di un ambiente di sistema e l'operazione, benché non difficile, richiede un minimo di attenzione e di conoscenza. L'argomento viene rimandato al paragrafo 19.8 nel capitolo 19.

12.5 La creazione sincrona dell'indice analitico e/o del glossario

Si è già detto altrove che le versioni moderne delle distribuzioni del sistema TEX usano come interprete il programma pdftex al quale viene associato un determinato file di formato, nel nostro caso il formato `latex.fmt` oppure `pdflatex.fmt`; a seconda di quali opzioni siano attive al momento della creazione del formato e/o al momento della compilazione, l'interprete può svolgere determinate azioni, in particolare può produrre in uscita il file `.dvi` oppure il file `.pdf`; può eseguire i comandi estesi del programma etex oppure può riferirsi solo ai comandi dell'in-

terprete tex; può chiedere al sistema di eseguire comandi esterni, oppure di non farlo.

In questo paragrafo viene esposta una soluzione al problema di eseguire immediatamente la conversione del file grezzo dell'indice analitico `.idx` nel file indicizzato e, volendo, la conversione del file grezzo del glossario `.glo` nel file indicizzato `.gls`. Questa operazione è possibile se l'interprete che si sta usando è abilitato per l'esecuzione di comandi di sistema.

Generalmente parlando questa operazione può venire eseguita a mano, in differita, cliccando sull'apposito bottone dello *shell editor* con cui si sta eseguendo la lavorazione del documento; normalmente non ci sono problemi, mentre per il glossario è necessario esprimere diverse opzioni al comando di sistema, per cui l'operazione, benché non sia difficile da eseguire dalla finestra comandi, come è stato descritto nei paragrafi precedenti, resta comunque una operazione suscettibile di errori. Per l'indice analitico l'operazione attraverso la finestra comandi diventa necessaria se si vogliono usare delle opzioni specifiche per il comando makeindex.

Ricordiamo innanzi tutto che l'indice analitico e il glossario vengono composti per ultimi, quando non è più necessario raccogliere informazioni da includere dentro questi elenchi. Ricordiamo anche che entrambi i file indicizzati hanno il nome identico a quello del file principale del documento che si sta componendo ⟨*job*⟩, e che questo nome è contenuto nella macro `\jobname`.

Allora è possibile eseguire la trasformazione immediata dei due file ⟨*job*⟩`.idx` e ⟨*job*⟩`.glo` subito dopo averli completati e subito prima di comporre ciascuna lista indicizzata ⟨*job*⟩`.ind` e ⟨*job*⟩`.gls`. Questo implica che bisogna chiudere immediatamente i due flussi di scrittura dell'interprete verso i file ⟨*job*⟩`.idx` e, rispettivamente, ⟨*job*⟩`.glo`. I nomi di questi due flussi di scrittura sono contenuti rispettivamente nelle due macro interne e protette di LaTeX `\@indexfile` e `\@glossaryfile`[1]. La chiusura di questi due flussi di scrittura deve essere istantanea, non differita alla prima occasione opportuna, come avviene invece normalmente per la scrittura su altri file.

Lo scrivente è ricorso all'uso del comando `\write18`, che permette di chiedere al sistema di eseguire l'argomento di questo comando, come se fosse un comando scritto nella finestra comandi, terminal, xterm, console, comunque si chiami nel sistema operativo che governa il calcolatore che si sta usando.

È bene notare che di default l'interprete pdftex, pur potendo eseguire il comando `\write18`, di default ne è disabilitato. Per abilitarlo è necessario scrivere

[1] Talvolta è necessario comporre indici analitici o glossari tematici; alcune classi non standard come *memoir* hanno un meccanismo interno che *non* consente di agire come descritto in questo e nei prossimi paragrafi. Altre classi non standard, come per esempio quelle della collezione KOMA-script, invece, generano inizialmente un solo file `.idx`, ma poi, grazie ad un applicativo esterno, spezzano questo file in un certo numero di file grezzi, con nomi distinti, ognuno dei quali va elaborato da makeindex. È possibile scrivere una serie di macro per gestire queste situazioni, e lasciare che i comandi descritti in questo e nei prossimi paragrafi, eseguano tutto il lavoro per ciascuno degli indici o glossari. È appunto quello che si è fatto per i cinque indici analitici di questo testo, sebbene la classe usata per la sua composizione sia la classe standard *book*.

l'apposita opzione nella linea di comando da inserire a mano da una finestra comandi, o da inserire nella configurazione dello *shell editor*. Per l'eseguibile pdftex.exe che compare nella distribuzione MiKTEX l'opzione da inserire nella linea di comando è `--enable-write18`; per gli eseguibili pdftex delle versioni derivate dalla distribuzione TEX Live l'opzione è `-shell-escape`. È importante configurare il proprio *shell editor* con questa opzione, altrimenti si rimane delusi, perché quanto descritto qui non potrebbe assolutamente funzionare.

Queste precisazioni sono necessarie, ma la situazione non è così complicata; infatti dal 2013 i programmi di composizione possono eseguire alcuni (pochi) comandi esterni senza essere esplicitamente abilitati a farlo mediante le opzioni `-shell-escape` o `--enable-write18`; fra questi pochi comandi liberamente eseguibili c'è makeindex, quindi i problemi relativi all'esecuzione di comandi esterni non ci sono. Inoltre esiste un pacchetto che permette di fare tutto il lavoro con poche impostazioni a livello del preambolo del documento che si sta compilando.

12.5.1 Indicizzazione sincrona dell'indice analitico

Qui si dà una prima descrizione elementare di come sarebbe necessario operare se non esistessero i pacchetti *imakeidxx* e *indextools*. Poi si vedrà come comportarsi con l'uso di questi pacchetti.

Per l'indice analitico, specialmente quando si vuole usare uno stile di composizione diverso da quello di default, i cui comandi sono contenuti nel file ⟨*stile*⟩`.ist`, basta inserire i comandi:

```
\maketaletter
\immediate\closeout\@indexfile
\immediate\write18{makeindex -s ⟨stile⟩.ist \jobname}
\makeatother
```

subito prima di

```
\input\jobname.ind
```

Quei comandi eseguono *immediatamente* la chiusura del flusso di scrittura che serve all'interprete per scrivere nel file `.idx` i lemmi dell'indice analitico. Altrettanto *immediatamente* lanciano una richiesta al sistema di eseguire il comando makeindex passandogli lo stile di composizione ⟨*stile*⟩`.ist` affinché con quello stile lavori sul file il cui nome è contenuto in `jobname` e la cui estensione vale di default `.idx`. In uscita viene prodotto un file con lo stesso nome, ma con l'estensione di default `.ind`. In questo modo quando viene eseguito il comando `\input` sul file `\jobname.ind`, la composizione dell'indice analitico viene eseguita sui dati appena raccolti e indicizzati, e non c'è più bisogno di ricorrere alle tre compilazioni come detto inizialmente.

Per questo testo è stato usato lo stile `classic.ist`, frutto di una collaborazione con Lorenzo Pantieri che usa uno stile simile per il suo testo [49]. I comandi inseriti nel file sorgente per indicizzare l'indice analitico e per introdurlo nel testo sono:

```
\makeatletter
\immediate\closeout\@indefile
\makeatother
\immediate\write18{makeindex␣-s␣classic.ist␣\jobname}
\input{\jobname.ind}
```

Al posto di `\jobname` si sarebbe potuto scrivere esplicitamente il nome del file principale `GuidaGuIT`, ma lasciando `\jobname` quei comandi possono essere usati senza nessuna modifica per la composizione di qualunque altro documento.

12.5.2 Indicizzazione sincrona del glossario

In modo del tutto analogo i comandi per comporre il glossario, anche prescindendo dall'uso di `\printglossary` possono essere i seguenti:

```
\makeatletter
\immediate\closeout\@glossaryfile
\makeatother
\immediate\write18{makeindex -s gglo.ist -o \jobname.gls \jobname.glo}
\input{\jobname.gls}
```

Ovviamente, al posto dello stile di composizione `gglo.ist`, quello di default quando si usa LaTeX, anche se bisogna specificarlo, si potrebbe usare un qualunque altro stile adatto allo scopo. Le cartelle di ogni distribuzione TeX sono piene di ogni sorta di stili da usare con makeindex, ma pochissimi sono destinati alla composizione dei glossari; generalmente, oltre a `gglo.ist`, che fa eccezione, essi hanno l'estensione `.gst`; nella distribuzione 2008 di TeX Live sono presenti i file `memman.gst`, usato per la composizione del glossario relativo alla documentazione della classe *memoir*, e `srcguide.gst`, relativo alla documentazione delle classi KOMA-script; benché scritti per quegli scopi, possono venire usati anche per altri glossari[2].

Non è particolarmente difficile crearsi un altro stile personalizzato; basta leggere accuratamente la documentazione che accompagna il programma makeindex in `.../doc/makeindex/` e prendere come modello i file di stile suggeriti in questo paragrafo.

12.6 Composizione automatica dell'indice analitico

I pacchetti *imakeidx* e *indextools* permettono di comporre gli indici analitici in modo 'sincrono' senza bisogno di fare nulla se non specificare qualche impostazione iniziale. Sono stati scritti per comporre gli indici analitici, ma si ritiene che siano usabili anche per i glossari; chi scrive non li ha mai provati in questo senso.

[2] Attenzione, però, perché la sintassi del contenuto dei comandi `\glossary` è in parte differente da quella prescritta dallo stile `gglo.ist`; per esempio, il carattere speciale `@` viene sostituito con `?`.

Caricato l'uno o l'altro pacchetto (qui si esemplifica con *imakeidx* ma quanto detto vale tale e quale con *indextools* che non è altro che in *fork* del precedente) nel solito modo, bisogna specificare di voler comporre uno o più indici:

```
\usepackage[⟨opzioni del pacchetto⟩]{imakeidx}
\makeindex[⟨opzioni di makindex⟩]
```

Nel corpo del testo i termini da indicizzare vengono introdotti con

```
\index[⟨opzioni per l'indicizzazione⟩]{⟨lemma⟩}
```

Verso la fine del documento, al momento di stampare l'indice o gli indici analitici, si scrive

```
\printindex[⟨opzioni per la stampa⟩]
```

Come si vede, a parte le opzioni, i comandi da usare sono gli stessi che si userebbero senza ricorrere a questi due pacchetti 'magici'.

Chiaramente le ⟨*opzioni per il pacchetto*⟩ sono globali e vanno scelta fra quelle esposte nelle rispettive documentazioni; spesso non è necessaria nessuna di queste opzioni, ma all'occorrenza esistono e sono accessibili.

Le ⟨*opzioni per makeindex*⟩ sono le più interessanti; quesi tutte sono della forma *chiave* = *valore*; qui se ne citano alcune:

nome = È il nome simbolico per identificare un particolare indice quando si compongono diversi indici; se non si specifica nulla viene prodotto un solo file grezzo `\jobname.idx`, altrimenti viene prodotto il file ⟨*nome*⟩`.idx`.

title = ⟨*titolo*⟩ dell'indice con il ⟨*nome*⟩ simbolico. Usando queste due chiavi non è difficile creare un indice identificato con `Nomi` che, quando sarà composto, invece del titolo di default "Indice analitico", avrà il titolo, per esempio, "Nomi dei personaggi importanti".

options = `{`⟨*opzioni*⟩`}` La lista delle ⟨*opzioni*⟩ da passare a programma di composizione dell'indice (di default il programma makeindex, per specificargli, per esempio, il file di stile da usare. Si noti che il 'valore' di questa opzione deve essere racchiuso fra parentesi graffe.

intoc è una chiave booleana che non necessita di specificarne il 'valore, che di default vae 'true'; bisogna specificare, eventualmente, solo il valore 'false' ma per ottenere questo risultato basta non specificare l'opzione' Serve per far si che il titolo del particolare indice sia inserito nell'indice generale.

columns = ⟨*numero*⟩ Serve per specificare come comporre l'indice analitico in colonne; di default ⟨*numero*⟩ vale 2, ma si può specificare 1, per esempio per un indice dei nomi o per un glossario; mentre si può specificare un ⟨*numero*⟩ maggiore di 2 per indici che usano lemmi molto corti.

Sono disponibili anche altre opzioni, ma si rinvia alla documentazione.

Per il comando `\index` è disponibile uns sola opzione, il ⟨*nome*⟩ che identifica il particolare indice in cui inserire il ⟨*lemma*⟩.

Analogamente il comando `\printindex` accetta come opzione solo il ⟨*nome*⟩ simbolico dell'indice che si vuole stampare. Se è necessario è possibile inserire fra il titolo dell'indice identificato da ⟨*nome*⟩ e l'indice vero e proprio un testo esplicativo composto a giustezza piena, non nella prima colonna dell'indice; questo testo si ottiene con il comando:

`\indexprologue[`⟨*spazio*⟩`]{`⟨*testo esplicativo*⟩`}`

Lo spazio viene inserito fra il titolo dell'indice e il ⟨*testo esplicativo*⟩. Composto l'indice, il ⟨*testo esplicativo*⟩ viene annullato, per cui se per un indice successivo occorre un altro ⟨*testo esplicativo*⟩ basta usare il comando senza dover preoccuparsi di 'cancellare' il testo precedente.

Gli indici analitici di questa guida sono stati composti usando questo metodo;anche se gli indici vengono ricomposti ogni volta che si ricompone il documento, il 'tempo perso' è complessivamente di un secondo o due, e compensa abbondantemente la necessità di ricordarsi di aggiornare l'indice analitico ogni volta che si fanno delle modifiche al testo o comunque anche quando il testo è pronto per essere licenziato e caricato nel sito GUIT.

Bibliografia

[1] Artech House. "Style Manual". Artech House, Boston – London.

[2] A.V. *The Chicago Manual of Style*. The University of Chicago Press, 14 ed., 1994.

[3] A.V. "Una (mica tanto) breve introduzione a LaTeX 2ε Ovvero LaTeX 2ε in 93 minuti". PDF document, 9 2000. In `$TEXMF/doc/latex/lshort-italian/itlshort.pdf`. Traduzione a cura di G. Agostini, G. Bilotta, F. Casadei Della Chiesa, O. de Bari, G. Delre, L. Ferrante, T. Pecorella, M. Rigido, R. Zanasi.

[4] Claudio Beccari. *Il LaTeX Refernece Manual commentato*. GuIt, `http://www.guitex.org/home/images/doc/GuideGuIT/`, 2015.

[5] Claudio Beccari. "La classe TOPtesi". PDF document, 11 2015. In `$TEXMF/doc/latex/toptesi/toptesi.pdf`, `$TEXMF/doc/latex/toptesi/toptesi-it-xetex.pdf`.

[6] Gian Luigi Beccaria. *Il mare in un imbuto*. Giulio Einaudi editore, Torino, 2010.

[7] Francesco Biccari. "Documentation of the LaTeX class sapthesis.cls". PDF document, 3 2012. In `$TEXMF/doc/latex/sapthesis/sapthesis-doc.pdf`.

[8] Robert Bringhurst. *The elements of typographic style*. Hartley & Marks, Vancouver, BC, 2004.

[9] Christophe Caignaert. "A story of *kpfonts*". In: *TGGboat*, vol. 31, no. 3, (2010), pp. 161–174.

[10] CEI 24-1. *Unità di misura e simboli letterali da usare in elettrotecnica*. Comitato Elettrotecnico Italiano, Milano, 1986.

[11] CEI-S-646. *Dizionario della strumentazione nucleare. Primo complemento al fascicolo S-447*. Comitato Elettrotecnico Italiano, Milano, 1983.

[12] Gustavo Cevolani. "Libretti in LaTeX". In: *ArsTEXnica*, vol. 1, no. 2.

[13] Gustavo Cevolani. "Norme tipografiche". In: *ArsTEXnica*, vol. 1, no. 1.

[14] CNR-UNI 10002. *Segni e simboli matematici per le scienze fisiche*

e tecniche. Unificazione Italiana, Milano, 1963.

[15] CNR-UNI 10003. *Sistema internazionale di unità (SI)*. Ente Italiano di Unificazione, Milano, 1984.

[16] Massimiliano Dominici. "Utilizzo di caratteri TrueType con LaTeX – Un esempio pratico: i *FellTypes*". In: *ArsTeXnica*, vol. 2, no. 4, (2007), pp. 88–102.

[17] Massimiliano Dominici. "Una panoramica su Pandoc". In: *ArsTeXnica*, vol. 8, no. 15, (2013), pp. 31–38.

[18] Michael Downes. *Short math guide*. American Mathematical Society, Providence, Rhode Island, 1.09 ed., 2002. In `ftp://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf`.

[19] Olaf Drümmer, Alexandra Oettler, Dietrich von Seggern. *PDF/A in a Nutshell. Long Term Archiving with PDF*. Callas Software gmbh, 2008. In: `http://www.pdfa.org/doku.php?id=pdfa:en:pdfa_in_a_nutshell`.

[20] Elsevier. "Preparing articles with LaTeX — Instructions to authors for preparing compuscripts". PDF document. In `instraut.pdf`.

[21] Giorgio Fioravanti. *Il manuale del grafico — Guida alla progettazione grafica e all'impaginazione del prodotto editoriale*. Zanichelli, Bologna, 1987.

[22] Peter Flynn. *Formatting information — A beginner's introduction to typesetting with LaTeX*. In `CTAN/tex-archive/info/beginlatex/beginlatex-3.6.pdf`.

[23] Michel Goossens. *The XeTeX Companion – TeX meets OpenType and Unicode*. LaTeX Team, gennaio 2011. In `http://xml.web.cern.ch/XML/lgc2/xetexmain.pdf`.

[24] Enrico Gregorio. "Installare TeX Live 2010 su Ubuntu". In: *ArsTeXnica*, vol. ottobre, no. 10, (2010), pp. 7–13.

[25] Enrico Gregorio. "L'arte esoterica di scrivere in cirillico con LaTeX". In: *ArsTeXnica*, vol. aprile, no. 9, (2010), pp. 57–73.

[26] Enrico Gregorio. "Come comporre un frontespizio e vivere felici". PDF document, 9 2011. In `$TEXMF/doc/latex/frontespizio/frontespizio.pdf`.

[27] Enrico Gregorio. *Introduzione a XeLaTeX*, gennaio 2011. In `http://profs.sci.univr.it/~gregorio/introxelatex.pdf`.

[28] ISO 690. *Documentation – Bibliographical references – Essential and supplementary elements*. International Organization for Standardization, Ginevra, 1982.

[29] ISO Handbook 2. *Units of measurement*. International Organization for Standardization, Ginevra, 1982.

[30] Kluver. "User manual for `kluwer.cls`". In `$TEXMF/doc/latex/kluwer/usrman.dvi`.

[31] Donald E. Knuth. *The TEXbook*. Addison Wesley, Reading, Mass., 16 ed., 1996.

[32] Donald E. Knuth. *Computers & typesetting*. Addison Wesley, Reading, Mass., 2001. Millenium edition.

[33] Helmut Kopka, Patrick W. Daly. *Guide to LATEX*. Addison Wesley, Reading, Mass., 4 ed., 2004.

[34] Steven G. Krantz. *A primer in mathematical writing*. American Mathematical Society, Providence, Rhode Island, 2 ed., 1998.

[35] Leslie Lamport. *A document preparation system — LATEX — User's guide and reference manual*. Addison Wesley, Reading, Mass., 2 ed., 1994.

[36] Jerónimo Leal, Gianluca Pignalberi. *Edizioni Critiche – Guida alla composizione con il proprio computer*. TEXnologie. Edizioni CompoMat, Configni (RI), 2012.

[37] Roberto Lesina. *Il nuovo manuale di stile*. Zanichelli, Bologna, 2 ed., 1994.

[38] André Miede. "A classic thesis style". PDF document, 12 2011. In `$TEXMF/doc/latex/classicthesis/classicthesis.pdf`.

[39] Michael Mitchel, Susan Wightman. *Book typography — A designer's manual*. Libanus Press, Marlborough, Wiltshire UK, 2005.

[40] Frank Mittelbach, Michel Goosens, *et al. The LATEX companion*. Addison Wesley, Reading, Mass., 2 ed., 2004.

[41] Frank Mittelbach, Michel Goosens, *et al. The LATEX graphics companion*. Addison Wesley, Reading, Mass., 2 ed., 2007.

[42] Bice Mortara Garavelli. *Prontuario di punteggiatura*. Editori Laterza, Bari, 11 ed., 2008.

[43] Bice Mortara Garavelli (cur.). *Storia della punteggiatura in Europa*. Editori Laterza, Bari, 2008.

[44] Imprimerie Nationale (cur.). *Règles typogrphiques*. Imprimerie Nationale, Parigi, 2006. ISBN 2-7433-0482-0.

[45] Scott Pakin. "The comprehensive LATEX symbol list". PDF document. In `$TEXMF/doc/latex/comprehensive/symbols-a4.pdf`.

[46] Lorenzo Pantieri. "LATEX per l'impaziente – Un'introduzione all'Arte di scrivere con LATEX". PDF document, settembre 2008. In `http://www.lorenzopantieri.net/LaTeX_files/LaTeXimpaziente.pdf`.

[47] Lorenzo Pantieri. "Tesi Classica". PDF document, 5 2012. In `http://www.lorenzopantieri.net/LaTeX_files/TesiClassica.zip`.

[48] Lorenzo Pantieri. "Tesi Moderna". PDF document, 5 2012. In `http:`

//www.lorenzopantieri.net/LaTeX_files/TesiModerna.zip.

[49] Lorenzo Pantieri, Tommaso Gordini. "L'arte di scrivere con LaTeX". PDF document, 2015. In http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf.

[50] Will Robertson. "The XeTeX reference guide". PDF document, gennaio 2011. In $TEXMF/doc/xetex/xetexref/XeTeX-reference.pdf.

[51] Francesca Serafini. *Questo è il punto – Istruzioni per l'uso della punteggiatura*. Editori Laterza, Bari, 2012.

[52] Fabrizio Serra. *Regole editoriali, tipografiche & redazionali*. Istituti Editoriali e Poligrafici Internazionali, Pisa – Roma, 2004. Prefazione di Martino Mardersteig, postfazione di Alessandro Olschki, appendice di Jan Tschichold.

[53] N. Setzer. "The cool package". PDF document. In $TEXMF/doc/latex/cool/cool.pdf.

[54] Societa Italiana di Fisica e Il Nuovo Cimento, Bologna. *Guida per gli autori*, 1988.

[55] Ellen Swanson, Arlene O'Sean, Antoinette Schleyer. *Mathematics into type*. American Mathematical Society, Providence, Rhode Island, 2 ed., 1999.

[56] Till Tantau. "Ti*k*Z and pgf — Manual for version 1.10". PDF document. In texlive/2012/texmf-dist/doc/generic/pdf/version-for-pdftex/en/pgfmanual.pdf.

[57] A. Thompson, B. N. Taylor. "The NIST Guide for the use of the International System of Units", 2008. In http://physics.nist.gov/cuu/pdf/sp811.pdf.

[58] Edward R. Tufte. *The visual display of quantitative information*. Grapphics Press, Cheshire, Connecticut, 2 ed., 2007.

[59] UNI 2949. *Diagrammi e cartogrammi, Regole generali per l'elaborazione*. Ente Italiano di Unificazione, Milano, 1982.

[60] UNI 6015. *Segnaccento obbligatorio nell'ortografia della lingua italiana*. Ente Italiano di Unificazione, Milano, 1967.

[61] UNI 6017. *Descrizioni e riferimenti bibliografici*. Ente Italiano di Unificazione, Milano, 1967.

[62] UNI 6461. *Divisione delle parole in fin di linea*. Ente Italiano di Unificazione, Milano, 1969.

[63] UNI 7090. *Metodo di scrittura numerica delle date*. Ente Italiano di Unificazione, Milano, 1973.

[64] UNI-ISO 5966. *Presentazione dei rapporti scientifici e tecnici*. Ente Italiano di Unificazione, Milano, 1989.

[65] Ivan Valbusa. "User's guide to suftesi". PDF document, 2015. In $TEXMF/doc/latex/suftesi/suftesi.pdf.

[66] Herbert Voss. "Math mode". PDF document, 2010. In $TEXMF/doc/latex/mathmode/.

[67] R.C. Weast, M.J. Astle, W.H. Beyer (cur.). *CRC handbook of chemistry and physics*, CRC Press, Boca Raton, Florida, cap. "Symbols, units and nomenclature in physics", pp. F259–F293. 65 ed., 1984.

[68] Peter Wilson. "The memoir class for configurable typesetting — User guide". PDF document, 2004. In `\protect\T1\textdollarTEXMF/doc/latex/memoir/memman.pdf`.

[69] Peter Wilson. *A few notes on book design*. The Herries Press, Normandy Park, WA, 2009.

Indice analitico

Simboli

!, 607, 676
\!, 285, 654
\", 635
", 290, 400
"<, 630
">, 630
#, 99
$, 99
%, 99
&, 99
\', 635, 681, 682
', 630
'', 630
\(, 263, 283, 651, 655, 669
\), 263, 283, 651, 655, 669
\+, 681
\,, 293, 630, 654
\-, 27, 334, 401, 559, 561, 566, 580, 681
-, 630
--, 630
---, 630
\., 635
\:, 290, 654
\;, 654
\<, 681
<, 169, 672
≪, 630
\=, 635, 681, 682
=, 672
\>, 681
>, 169, 672
≫, 630
@, 236, 400, 405, 416
\@, 630
@-espressione, 156, 683
\@dottedtocline, 407, 416
\@evenhead, 466
\@glossaryfile, 246
@gobble, 414
\@ifdefinable, 128
\@ifundefined, 128
\@indexfile, 246
\@oddhead, 466
\@Roman, 406, 409
\@roman, 405, 408
\@sptoken, 411
\@tfor, 412
\[, 253, 264, 290, 651
\~, 635
\\, 143, 144, 157, 159, 164, 281, 401, 602, 627–629, 643, 644, 681, 683, 684, 690
*, 281, 629, 690
\^, 635
\], 253, 264, 290, 651
^, 99
\|, 257, 258
_, 99
\‘, 635, 681, 682
‘, 414, 630
‘‘, 630
{, 99
}, 99

~, 400, 602, 630, 644
␣, 630

A

\a', 682
\a=, 682
\a‘, 682
\AA, 635
\aa, 635
\abovedisplayshortskip, 652
\abovedisplayskip, 652
\accent, 559, 589
\addcontentsline, 422, 637
\addconttentsline, 637
\addto, 404, 560
\addtocounter, 627, 659
\addtolength, 663
\addvspace, 673
\advance, 664
\AE, 635
\ae, 635
\AfterEndPreamble, 617
\afterpage, 167, 168, 176, 427, 428, 607
\aleph, 258
\allowdisplaybreak, 281
\allowdisplaybreaks, 280, 281
\Alph, 660
\alph, 660
\alpha, 253, 254
\amalg, 255
\and, 644, 669
\angle, 258
\ap, 266, 652
\appendix, 108, 118
\approx, 255
\arabic, 659, 661
\arc, 182
\arc*, 182
\arccos, 256
\arcsin, 256
\arctan, 256
\arg, 256
\arraybackslash, 169
\arraycolsep, 684
\arrayrulewidth, 684
\arraystretch, 160, 684
\Arrowvert, 257
\arrowvert, 257
\ast, 255
\asymp, 255
\AtBeginDocument, 411, 670
\Author, 494
\author, 329, 469, 629, 644
\autore, 469

B

\b, 635
\backmatter, 108, 112, 118, 466, 636
\backslash, 257, 258
\baselineskip, 423, 455, 632, 691
\beamertemplatetransparentcovereddynamic, 329
\begin, 105, 132–134, 136, 137, 139, 142, 154, 156, 175, 179, 264, 290, 418, 472, 568, 614, 629, 645, 649–651, 657, 674, 681, 682, 686–688, 693, 695, 696
\begingroup, 142
\belowdisplayshortskip, 652
\belowdisplayskip, 652
\beta, 253, 254
Beta Code, 300
\bfseries, 171, 349, 358, 705
\bgroup, 142, 419, 697
\bibindent, 639
\bibitem, 139–142, 628, 686
\bibliography, 232, 238, 685
\bibliographystyle, 232
\bigbreak, 692
\bigcap, 256
\bigcirc, 255
\bigcup, 256
\Biggl, 262
\biggl, 262
\Biggr, 262

\biggr, 262
\Bigl, 262
\bigl, 262, 277
\bigodot, 256
\bigoplus, 256
\bigotimes, 256
\Bigr, 262
\bigr, 262, 277
\bigskip, 673
\bigskipamount, 692
\bigsqcup, 256
\bigtriangledown, 255
\bigtriangleup, 255
\biguplus, 256
\bigvee, 256
\bigwedge, 256
\binom, 282
\bm, 286, 363
\boldmath, 286, 361, 655
\boldsymbol, 286, 363
\boolean, 668
\bot, 258
\botfigrule, 176, 674
\botmark, 465, 467
\bottomfraction, 677, 679
\bottomrule, 161
\bowtie, 255
\Box, 258
\box, 420, 698, 699
\boxed, 287
\bracevert, 257
\break, 334
\bullet, 255
\buonfinepagina, 609

C

\c, 635
\cap, 255
\capolettera, 402
\capoverso, 402
\caption, 146, 155, 168, 172, 635, 636, 676, 677, 699
\cbezier, 181, 182
\cdot, 255
\cdots, 653
\centering, 142–145, 169, 629, 683
\cfrac, 282
\chapter, 111, 112, 423, 465, 466, 520, 635, 636
\chapter*, 96, 637
\chaptermark, 466
\chaptername, 635, 636
\char, 401, 589
\chi, 254
chiave
  angle, 227
  bb, 225
  clip, 227
  height, 226
  keepaspectratio, 226
  trim, 227
  viewport, 226
  width, 226
\circ, 255
\circle, 180
\circle*, 180
\cite, 140–142, 233, 312, 503, 686
\cleaders, 424, 426
\cleardoublepage, 669, 676, 691
\clearpage, 176, 601, 605, 607, 676, 687, 691
\clerapage, 607
\cline, 159, 160, 289, 684
\closeout, 247
\clubpenalty, 609
\clubsuit, 258
codice di allineamento
  B, 702
  b, 157, 682, 696, 702
  c, 157, 682, 696
  l, 693, 702
  r, 693, 702
  s, 693
  t, 157, 682, 696, 702
colophon, 469
\color, 704
\colorbox, 704
\columnsep, 632
\columnseprule, 632

\columnwidth, 226, 632, 664
\cong, 255
contatore
  bottomnumber, 677–679
  dbltopnumber, 678, 679
  footnote, 633
  MaxMatrixCols, 287
  secnumdepth, 635–637
  table, 172
  tocdepth, 635–637
  topnumber, 677–679
  totalnumber, 677–679
\coprod, 256
\copyright, 635
\cos, 256
\cosh, 256
\cot, 256
\coth, 256
\cs, 401
\csc, 256
\csname, 434, 649
\cspreto, 318
\cup, 255

D

\D, 161
\d, 635
\dag, 635
\dagger, 255
\dashbox, 180, 181
\dashv, 255
\date, 329, 469, 629, 644
\dbinom, 282
\dblfigrule, 176, 674
\dblfloatpagefraction, 678, 679
\dblfloatsep, 678, 679
\dbltextfloatsep, 678, 679
\dbltopfraction, 678, 679
\ddag, 635
\ddagger, 255
\ddots, 653
\DeclareGraphicsRule, 212, 220
\DeclareMathOperator, 259, 281, 285
\DeclareMathOperator*, 259
\DeclareMathSizes, 705
\DeclareMathSymbol, 411, 617
\DeclareOperatorName, 259
\DeclareRobustCommand, 417
\def, 128, 161
\definecolor, 703
\deg, 256
\Delta, 253, 254
\delta, 254
\depth, 693
descrittore di colonna
  @, 156, 683
  |, 156, 683
  b, 159, 164, 628
  C, 171
  c, 156, 171, 683
  D, 168
  E, 171
  L, 171
  l, 156, 171, 683
  m, 159, 164, 628
  p, 156, 159, 164, 628, 683, 684
  r, 156, 171, 683
  S, 164
  s, 164
  X, 164
\det, 256
\dfrac, 282, 283
\Diamond, 258
\diamond, 255
\diamondsuit, 258
\diff, 285
\dim, 256
\dimexpr, 663, 665
\displaybreak, 280, 281
\displaystyle, 172, 252, 492, 656
\div, 255
\divide, 664, 666
\do, 412
\documentclass, 93, 96, 104, 122, 167, 352, 629, 638, 687
\doteq, 255
\dotfill, 424, 426
\dots, 653, 654

\doublerulesep, 684
\Downarrow, 255, 257
\downarrow, 255, 257
\dp, 700

E

\edef, 128
\egroup, 142, 419
\ell, 258
\else, 411, 412, 616
\em, 359, 560
\emph, 75, 172, 359, 560, 631
\emptyset, 258
\end, 95, 105, 128, 133, 134, 136, 137, 139, 142, 154, 156, 175, 179, 264, 418, 472, 568, 614, 629, 645, 650, 651, 657, 674, 676, 681, 682, 686–688, 693, 695, 696
\endcsname, 434, 649
\endfirsthead, 172
\endfoot, 172
\endgroup, 142
\endhead, 172
\endinput, 126
\endlastfoot, 172
\endlist, 649
\endmulticols, 423
\enlargethispage, 457, 608, 633, 691, 692
\enlargethispage*, 633, 691
\enspace, 171, 290
\ensuremath, 651
\epsilon, 254
\eqref, 265
\equal, 667
\equiv, 255
\ergoup, 697
\errorstopmode, 620
\eta, 254
\etichettadescrizione, 650
\evensidemargin, 643
\exists, 258, 272
\exp, 256
\expandafter, 412, 414, 621, 649
\extracolsep, 162, 173, 683

F

\f@baselineskip, 405
\fbox, 430, 693, 695, 696, 704
\fboxrule, 419, 693, 704
\fboxsep, 419, 693, 704
\fcolorbox, 704
\fi, 411, 412, 616
\figurename, 635, 636
\filbreak, 609, 692
file
  di formato, 79
  di inizializzazione, 81
\fill, 162, 662, 683
\finalhyphendemerits, 334, 601
\firstmark, 465, 467
\flat, 258
\floatpagefraction, 677–679
\floatsep, 678, 679
\fnsymbol, 660
\fontencoding, 350
\fontseries, 390
\fontshape, 390
\fontsize, 406
\footnote, 145, 146, 503, 504, 558, 628, 633
\footnotemark, 146, 147, 633
\footnoterule, 633, 674
\footnotesep, 633
\footnotesize, 165, 166, 353, 446, 705
\footnotetext, 146, 147, 628, 633
\footskip, 459, 460
\forall, 258
\frac, 282, 653
\frakfamily, 306
\framebox, 180, 181, 420, 430, 693
\frenchspacing, 477, 630, 631
\frontmatter, 108, 112, 118, 466, 636, 637
\frown, 255

\fussy, 690

G

\Gamma, 253, 254
\gamma, 253, 254
\gcd, 256
\gdef, 128, 411
\ge, 255
\geq, 255
\gets, 255
\gg, 255
\glossary, 244, 245, 638, 685, 689
\glossaryentry, 685
\glueexpr, 663
\goodbreak, 609, 692
\goodpagebreak, 609
\gothfamily, 306
\graphicspath, 222–224
\greekfont, 315
\GuIT, 94

H

\H, 635
\halign, 173
\hangafter, 403
\hangindent, 403
\hat, 99
\hbadness, 565
\hbar, 258
\hbox, 406, 431, 695, 697, 698
\hbox to, 698
\header, 459
\headsep, 459, 460
\heartsuit, 258
\height, 693
\hfill, 425, 468, 673
\hfuzz, 565
\hline, 159, 684
\hom, 256
\hookleftarrow, 255
\hookrightarrow, 255
\hrule, 172, 425, 433
\hrulefill, 424, 425
\hsize, 419
\hskip, 424, 559
\hspace, 673
\hspace*, 673
\hss, 426, 649, 698
\ht, 700
\Huge, 353, 700, 705
\huge, 353, 403, 705
\hyphenation, 26, 27, 562, 564, 568, 627
\hz, 560

I

\i, 505, 587, 635
\idotsint, 284
\if, 412
\ifbool, 671
\ifcat, 414
\ifcsdef, 671
\ifcsempty, 671
\ifcsequal, 671
\ifcsname, 414
\ifcsprefix, 671
\ifcsstring, 672
\ifcsundef, 671
\ifdef, 671
\ifdefempty, 671
\ifdefequal, 671
\ifdefprefix, 671
\ifdefstring, 672
\iff, 255
\ifFamily, 313
\iflanguage, 415
\iflettere, 409
\ifmmode, 671
\ifnum, 413, 616, 661
\ifnumcomp, 672
\ifnumequal, 672
\ifnumgreater, 672
\ifnumless, 672
\ifnumodd, 672
\ifodd, 428, 661, 671, 672
\ifthen, 673
\ifthenelse, 667, 669

\ifundef, 671
\ifx, 411–413
\iiiint, 284
\iiint, 284
\iint, 284
\Im, 258
\imath, 258, 262
\immediate, 247
\in, 255, 272
\include, 101–106, 614, 685, 687, 691
\includegraphics, 101, 191, 220–223, 225, 227, 330, 431, 433, 511, 679, 703
\includegraphics*, 225, 227
\includeonly, 101–103, 105, 106, 108, 685, 687
\includesvg, 222
\indent, 144, 631
\index, 241, 242, 245, 249, 250, 401, 638, 641, 685, 688, 689
\indexentry, 242, 685
\indexname, 422
\indexprologue, 250
\indexspace, 421
\indici, 118
\inf, 256
\infty, 258, 414
\input, 101–104, 106, 168, 243, 247, 687, 688
\insertpenalties, 606
\institution, 329
\int, 256, 284
\IntelligentComma, 132
interprete TEX, 83
\intertext, 282, 283, 631, 704
\intextsep, 678, 679
\iota, 254
\isodd, 668
\isundefined, 668
\it, 357
\item, 136, 137, 139, 330, 423, 559, 627, 645, 646
\itemindent, 646
\itemsep, 646, 649
\itshape, 171, 349, 358, 705

J

\j, 635
\jmath, 258, 262
\jobname, 246–248, 685
\Join, 255
\jot, 652
\justify, 142

K

\kappa, 254
\ker, 256
\Keywords, 494
\kill, 681

L

\L, 635
\l, 635
\l@chapter, 408
\l@figure, 416
\l@table, 416
\label, 138, 141, 146, 149, 155, 264, 265, 273, 280, 633, 647, 652, 657, 659, 677, 686
\labelsep, 646
\labelwidth, 646
\Lambda, 254
\lambda, 254
\langle, 257
\LARGE, 353, 705
\Large, 353, 705
\large, 353, 705
larghezza del font, 336
\LaTeX, 631
\LaTeXe, 631
\lceil, 257
\ldots, 653
\le, 255
leader, 424
\leaders, 424, 426
\leadsto, 255

\leavevmode, 425
\left, 261, 262, 277, 361
\Leftarrow, 255
\leftarrow, 255
\lefteqn, 652
\leftharpoondown, 255
\leftharpoonup, 255
\lefthypenmin, 568
\lefthyphenmin, 568
\leftmargin, 646, 649
\leftmargini, 646, 649
\leftmarginii, 646
\leftmarginiii, 646
\leftmarginiv, 646
\leftmark, 465–467
\Leftrightarrow, 255
\leftrightarrow, 255
legature e segni diacritici, 337
\lemma, 467, 468
\lengthtest, 667
\leq, 255
\let, 414, 649
\let@token, 411
\lfloor, 257
\lg, 256
\lgroup, 257
\lhd, 255
\lim, 256
\liminf, 256
\limsup, 256
\Line, 182
\line, 180
linea guida, 424
\linebreak, 334, 627, 690
\linepenalty, 603
\lineskip, 605, 632
\lineskiplimit, 632
\linespread, 356, 357, 632
\linethickness, 181
\linewidth, 166, 226, 419, 632, 696
\list, 649
\listfigurename, 404
\listfiles, 688
\listoffigures, 177, 404, 635
\listoftables, 177, 635
\listparindent, 646
\ll, 255
\llap, 403, 649
\lmoustache, 257
\ln, 256
\log, 256
\logo, 329
\long, 656, 671
\Longleftarrow, 255
\longleftarrow, 255
\Longleftrightarrow, 255
\longleftrightarrow, 255
\longmapsto, 255
\Longrightarrow, 255
\longrightarrow, 255
\looseness, 457, 608, 692
\LTleft, 172
\LTright, 172

M

\m@thcomma, 411
\MacTeX, 401
\maimatter, 636
\mainmatter, 108, 112, 118, 466, 636
\makeatletter, 405
\makeatother, 247
\makebox, 166, 180, 181, 185, 430, 693, 695, 698
\makeglossary, 244, 638, 685, 689
\makeindex, 241, 249, 638, 685, 688
\makelabel, 646
\MakeLowercase, 308
\maketaletter, 247
\maketitle, 469, 644, 660
\MakeUppercase, 308, 422, 466
\mapsto, 255, 272, 491, 492
\marginpar, 147, 148, 633, 679
\marginparpush, 634, 680
\marginparsep, 634, 680
\marginparwidth, 634, 680
\markboth, 423, 465–467, 637, 642
\markright, 423, 465–467, 637, 642
\masthindent, 640

\mathbb, 271
\mathbf, 286, 655
\mathcal, 290, 361, 655
\mathchoice, 491, 492
\mathindent, 640, 652
\mathit, 254, 364, 655
\mathnormal, 492
\mathop, 280, 285
\mathord, 411
\mathpunct, 411
\mathr, 263
\mathrel, 492
\mathrm, 263, 655, 704
\mathsf, 364, 655
\mathtt, 364, 655
\max, 256
\maxdimen, 620, 665
\mbox, 263, 430, 631, 693, 694, 697, 698
\mdseries, 349, 358, 705
\medbreak, 692
\medskip, 172, 673
\medskipamount, 692
METAPOST, 89
\mho, 258
\mid, 255
\middle, 361
\midrule, 161
\min, 256
\mkern, 654
\models, 255
\mp, 255
\mskip, 654
\mu, 254
\muexpr, 663
\multicolumn, 158, 684
\multiply, 664, 666
\multiput, 179, 181

N

\nabla, 258
\natural, 258
\ne, 255
\nearrow, 255
\NeedsTeXFormat, 126
\neg, 258
\neq, 255, 491
\newbool, 671
\newboolean, 668
\newbox, 698
\newcolumntype, 170, 171, 683
\newcommand, 127, 128, 280, 285, 400, 402, 404, 416–418, 424, 656, 690
\newcommand*, 400, 467
\newcount, 672
\newcounter, 627, 658, 672
\newenvironment, 418, 420, 421, 657, 698
\newfloat, 472
\newif, 668, 671
\newlength, 627, 663
\newline, 157, 629, 690
\newpage, 457, 609, 669, 691
\newsavebox, 430, 627, 694, 695, 698
\newtheorem, 117, 627, 657
\nexists, 272
\ni, 255
\noalign, 172
\nobreak, 334, 560
\nocite, 233, 238, 239
\noexpand, 414
\nofiles, 685
\noindent, 144, 419, 631
\NoIntelligentComma, 132
\nolinebreak, 627, 690
\nonfrenchspacing, 631
\nonumber, 652
\nopagebreak, 627, 691
\normalbaselineskip, 455
\normalfont, 357, 705
\normalmarginpar, 634, 679, 680
\normalsize, 353, 705
\not, 255, 272, 491, 669
\notag, 274
\notbool, 671
\notin, 272
\nu, 254

\null, 193, 468
\numberline, 638
numerazione
  Alph, 642, 660
  alph, 642, 660
  arabic, 642, 659
  fnsymbol, 642, 660
  Roman, 642, 660
  roman, 642, 659
\numexpr, 663, 672
\nwarrow, 255

O

\O, 635
\o, 635
occhiello, 469
\oddsidemargin, 643
\odot, 255
\OE, 635
\oe, 635
\ohm, 266, 417
\oint, 256
\oldstylenums, 364
\Omega, 254, 266
\omega, 254
\ominus, 255
\onecolumn, 643
\oplus, 255
opzione
  10pt, 446, 456, 638, 648
  11pt, 352, 356, 446, 639, 705
  12pt, 352, 446, 639
  a-1b, 493
  a4paper, 639
  a5paper, 639
  ansinew, 65, 95, 579
  applemac, 95, 579, 580, 585, 591
  b5paper, 639
  babelshorthands, 27
  backend, 240
  backref, 240
  charter, 365
  columns, 249
  config, 339
  cp1250, 580
  cp1252, 579
  cp437, 580
  draft, 639
  dvipdfm, 510
  enable-write18, 217
  EU1, 318
  executivepaper, 639
  final, 639
  fleqn, 640, 652
  garamond, 365
  greek, 299, 300
  hyperref, 240
  ibycus, 300
  intoc, 249
  italian, 122, 132, 133, 267, 290, 328, 410, 505, 652, 653
  landscape, 322, 639
  latin, 307
  latin1, 68, 95, 509, 510, 579, 580, 591, 593, 597, 600
  latin9, 95, 510, 579, 580
  legalpaper, 639
  leqno, 640
  letterpaper, 639
  LGR, 311, 313, 571
  LMR, 313
  ltxarrows, 178
  LY1, 377
  main, 367
  natbib, 240
  nome, 249
  notitlepage, 639
  oldstyle, 305
  onecolumn, 639
  oneside, 639
  openany, 639
  openbib, 639
  openright, 639
  options, 249
  origin, 702
  original, 178
  OT1, 95, 305, 589, 590
  OT2, 367, 368
  pdfa, 493

polutoniko, 301
pstarrows, 178
shell-escape, 217
square, 240
style, 239
T1, 94, 95, 305, 313, 318, 328, 367, 377, 388, 389, 570, 572, 581, 585, 588–590, 598
T2, 377
T2A, 367, 369
T2B, 367
T2C, 367
title, 249
titlepage, 639, 643
TS1, 377, 581
twocolumn, 639
twoside, 639, 642
utf8, 59, 60, 66, 68, 94, 95, 303, 309, 328, 369, 503, 571, 572, 580, 581, 590–593, 597, 598, 600
utopia, 365
veryoldstyle, 305
x-1a, 493
X2, 367–369
\or, 669
\Org, 494
\oslash, 255
\otimes, 255
\oval, 180, 181
\overbrace, 262
\overline, 654

P

\P, 635
\pagebreak, 281, 601, 627, 691
\pagecolor, 627, 704
\pagenumbering, 627, 642
\pageref, 139, 141, 155, 265, 652, 686
\pagestyle, 641
\paperheight, 665
\paperwidth, 665
\par, 143, 144, 402, 419, 423, 632, 656
\paragraph, 293, 635, 636
\parallel, 255
\parbox, 193, 430, 433, 656, 683, 696, 698
\parindent, 632
\parsep, 646, 648
\parskip, 605, 632
\part, 112, 635, 636
\partial, 258
\partname, 635, 636
\partopsep, 646
\pdfoutput, 83, 337
\ped, 266, 653
\perp, 255
\pgfdeclareimage, 329
\pgfuseimage, 329, 330
\Phi, 254
\phi, 254
\Pi, 254
\pi, 254
piedino, 17
\pm, 255
\pmb, 286
\polygon, 182, 183
\polygon*, 182, 183
\polyline, 182, 183
\poptabs, 682
posizione degli oggetti flottanti
  !, 154, 176, 674
  b, 154, 176, 674
  H, 176
  h, 154, 176, 674, 675
  p, 154, 176, 674, 677
  t, 154, 176, 674, 675
\pounds, 635
\Pr, 256
\prec, 255
\preceq, 255
\pretolerance, 335, 425, 603
\prime, 258
\printglossary, 244, 248
\printindex, 243, 244, 249, 250, 476, 688

\prod, 256
\propto, 255
\protect, 417, 628, 638, 671
protrusione dei segni grafici, 336
protrusione della punteggiatura, 336
\providebool, 671
\provideboolean, 668
\providecommand, 416, 420, 656
\providecommand*, 416
\ProvidesClass, 127
\ProvidesPackage, 126
\ProvideTextCommandDefault, 586
\Psi, 254
\psi, 254
\pushtabs, 681
\put, 179–182, 191

Q

\qbezier, 181, 182
\qquad, 276, 293, 346, 654
\quad, 276, 293, 346, 654

R

\r, 635
\raggedleft, 142, 148, 157, 169, 629, 680, 683
\raggedright, 142, 148, 157, 169, 629, 683
\raisebox, 403, 697
\rangle, 257
\rBrace, 192, 193
\rceil, 257
\Re, 258, 259
\ref, 139, 141, 149, 155, 265, 633, 647, 652, 657, 659, 686
\reflectbox, 703
\refstepcounter, 659, 686
refuso, 21
\relax, 128, 405, 616, 663, 664, 672, 674
\renewcommand, 127, 128, 160, 404, 405, 420, 634, 656, 678, 684
\renewcommand*, 404
\renewenvironment, 420, 421, 657
\RequirePackage, 428
\resizebox, 166, 702
\reversemarginpar, 147, 634, 679, 680
\rfloor, 257
\rgroup, 257
\rhd, 255
\rho, 254
\right, 261, 262, 277, 361
\Rightarrow, 255
\rightarrow, 255
\rightharpoondown, 255
\rightharpoonup, 255
\righthyphenmin, 568
\rightleftharpoons, 255
\rightmargin, 646
\rightmark, 465–467
risposta all'errore
  e, 109
  h, 109
  i, 109
  q, 109
  s, 109
  x, 109
\rlap, 492
\rm, 357
\rmfamily, 348, 357, 358, 705
\rmoustache, 257
\Roman, 660
\roman, 405, 659
\romannumeral, 409
\rotatebox, 702
\rule, 160, 161, 433, 492, 697

S

\S, 635
\savebox, 430, 695
\sbox, 430, 694, 698
\scalebox, 702
\scriptscriptstyle, 252, 492, 656
\scriptsize, 263, 353, 705

\scriptstyle, 252, 492, 656
\scshape, 349, 358, 705
\searrow, 255
\sec, 256
\section, 75, 112, 423, 465, 466, 520, 635, 636
\section*, 637
\selectfont, 350, 390, 406, 632
\selectlanguge, 404
\setactivedoublequote, 132, 290
\setbeamercolor, 329
\setboolean, 668
\setbox, 419, 695, 698, 699
\setcounter, 287, 627, 637, 658, 659, 678
\setISOcompliance, 132
\setlength, 178, 640, 663, 678
\setminus, 255
\settodepth, 663, 700
\settoheight, 663, 700
\settowidth, 663, 700
\sf@size, 405
\sfdefault, 318
\sffamily, 348, 358, 705
\sharp, 258
shell editor, 62
\shipoutAnswer, 150
\shorthandoff, 290, 291
\shorthandon, 290, 291
\shortstack, 629
\showboxbreadth, 619, 620
\showboxdepth, 619, 620
\showhyphens, 558, 564
\SI, 267
\Sigma, 254
\sigma, 254
\sim, 255
\simeq, 255
\simulatedCS, 410
\simulatedSC, 406, 408
\sin, 256
\sinh, 256
\sloppy, 690
\slshape, 349, 358, 705
\small, 165, 353, 446, 705
\smallbreak, 691
\smallskip, 172, 673
\smallskipamount, 691
\smile, 255
\spadesuit, 258
spazieggiatura, 336
\special, 79
\sqcap, 255
\sqcup, 255
\sqrt, 256, 653
\sqsubset, 255
\sqsubseteq, 255
\sqsupset, 255
\sqsupseteq, 255
\ss, 635
\stackrel, 654
\star, 255
\stepcounter, 659
stile della pagina
  empty, 423, 642
  headings, 423, 466, 642
  myheadings, 423, 466, 642
  plain, 423, 642
\stretch, 662
\stretchandshrink, 173
\string, 414
\strut, 159, 160, 629, 633
\subparagraph, 111, 635, 636
\subsection, 465, 520, 635, 636
\subset, 255
\subseteq, 255
\substack, 284
\subsubsection, 520, 635, 636
\subtitle, 329
\succ, 255
\succeq, 255
\sum, 256
\sup, 256
\supaginadispari, 428
\supaginapari, 428
\suppressfloats, 627, 677
\supset, 255
\supseteq, 255
\surd, 258
\swabfamily, 306

\swarrow, 255
\symbol, 706

T

\t, 635
\T1/lmr/m/n/10, 390
\tabcolsep, 164, 165, 684
\tablename, 635, 636
\tableofcontents, 635
\tabucline, 289
\tabularimage, 433, 434
\tabularnewline, 159, 164
\tabulinestyle, 289
\tag, 265, 274
\tan, 256
\tanh, 256
\tau, 254
testatina, 17
\testlettere, 409, 410
\TeX, 401, 631
\text, 263, 279, 283, 631, 704
\textasciicircum, 100
\textasciitilde, 100
\textbackslash, 100, 401
\textbf, 286, 349, 358, 705
\textcelsius, 504
\textcent, 586
\textcolor, 704
\textcurrency, 586
\textcyrillic, 368
\textdegree, 504, 586
\textdollar, 100
\texteuro, 171, 377
\textfloatsep, 678, 679
\textfraction, 677, 679
\textfrak, 306
\textgoth, 306
\textheight, 226, 455, 643, 665
\textit, 349, 655, 705
\textmd, 349, 705
\textmho, 366
\textnormal, 705
\textnumero, 34
\textohm, 266
\textormath, 417, 651
\textrm, 348, 357, 358, 631, 705
\textsc, 349, 405, 406, 705
\textsf, 348, 358, 705
\textsl, 349, 358, 705
\textstyle, 190, 252, 492, 656
\textsuperscript, 653
\textswab, 306
\texttt, 348, 401, 705
\textunderscore, 100
\textup, 349, 705
\textvisiblespace, 578
\textwidth, 226, 446, 632, 643, 664, 665
\tfrac, 282
\thanks, 628, 644, 660
\the, 657, 658, 661, 701
\thepage, 406
\Theta, 254
\theta, 254
\thetable, 172
\thicklines, 181
\thinlines, 181
\thinspace, 290
\thispagestyle, 423, 627, 641
thumbnail, 78
\times, 255
\tiny, 353, 705
\Title, 494
\title, 329, 469, 629, 643
\titlepage, 330
\titolo, 469
\to, 255
\today, 631
\tolerance, 335, 425, 568, 603
\top, 258
\topcaption, 168
\topfigrule, 176, 674
\topfraction, 677, 679
\topmargin, 643
\topmark, 465, 467
\toprule, 161
\topsep, 646, 649
\topskip, 455, 456
\totalheight, 693

\traceoff, 619, 620, 625
\traceon, 619, 620, 625
\tracingassigns, 618
\tracingcommands, 618, 619
\tracinglostchars, 618
\tracingmacros, 618, 619
\tracingonline, 618
\tracingoutput, 618
\tracingpages, 618, 622
\tracingparagraphs, 618, 622
\tracingrestores, 619
\tracingstats, 618
\triangle, 258
\ttfamily, 348, 358, 705
\twocolumn, 421, 627, 643
\twocolumns, 675
\typein, 628, 689
\typeout, 628, 689, 690

U

\U, 161, 257
\u, 257, 635
\uimm, 259, 280
\uishape, 357, 358
\unboldmath, 286, 361, 655
\underbrace, 262
\underline, 654
\unhbox, 699
\unhcopy, 699
\unit, 132, 265–267
unità di misura
  bp, 662
  cc, 662
  cm, 662
  dd, 662
  em, 662
  ex, 662
  in, 662
  mm, 662
  pc, 662
  pt, 662
  sp, 662
unità di misura, 10
  big point, 10, 662
  cicero, 12
  pica, 12
  punto didot, 12
  punto PostScript, 10
  punto tipografico anglosassone, 10
\unita, 267
\unitlength, 178, 179, 193
\unitre, 381
\unlhd, 255
\unrhd, 255
\unvbox, 420, 699
\unvcopy, 699
\Uparrow, 255, 257
\uparrow, 255, 257
\uplus, 255
\upshape, 349, 358, 705
\Upsilon, 254
\upsilon, 254
\usapackage, 428
\usebox, 430, 695, 699
\usecounter, 647
\useoutertheme, 329
\usepackage, 94, 121, 126, 141, 178, 249, 405, 640, 687
\usetheme, 328, 331

V

\V, 161, 257
\v, 635
\value, 428, 659, 661, 664, 667
\varDelta, 257
\varepsilon, 254
\varGamma, 257
\varphi, 254
\varpi, 254
\varrho, 254
\varsigma, 254
\varTheta, 257
\vartheta, 254
\vbadness, 566
\vbox, 419, 431, 698
\vbox to, 698
\vcenter, 431, 698

\vcenter to, 698
\vdash, 255
\vdots, 653
\vector, 180
\vee, 255
\verb, 650
\verb*, 650
\vert, 257
\vfill, 673
\vfuzz, 566
\virgola, 411, 412, 414, 617
virgola intelligente, 132
\virgoladecimale, 411, 412, 414
\vline, 156, 683
\vspace, 144, 402, 423, 673
\vspace*, 144, 673
\vsplit, 699
\vss, 698
\vtop, 431, 698, 699
\vtop to, 698

W

\wd, 700
\wedge, 255
\whiledo, 471, 669
\widowpenalty, 609
\width, 472, 693
\wp, 258
\wr, 255
\write18, 246, 247

X

\xdef, 128
\Xi, 254
\xi, 254
\xleaders, 424, 426
\xleftarrow, 284
\xrightarrow, 284

Z

\z@, 560
\zeta, 254

Indice degli ambienti

A

abstract, 112, 644
align, 277, 278, 280, 293
alignat, 279
aligned, 273–275, 280
alignedat, 280
alltt, 640, 650
amsmath, 276
Answer, 149
array, 159, 164, 289, 628, 629, 682

B

Bmatrix, 287
bmatrix, 287

C

cases, 261, 283
CD, 290
center, 142, 144, 145
checkhyphens, 558

D

description, 139, 645, 649
descrizione, 650
displaymath, 253, 264, 265, 274, 651
document, 96, 167, 501

E

enumerate, 136, 647, 649
eqnarray, 273, 274, 629, 651, 652
*eqnarray**, 273, 651
equation, 264, 274, 277, 651, 652
*equation**, 264, 265, 274
Exercise, 149

F

figure, 144, 175, 177, 178, 221, 471, 677
filecontents, 629, 688
flalign, 278
flushleft, 142, 508
flushright, 142
frame, 324, 330

G

gather, 277
gathered, 280
glossary, 638

I

index, 638
itemize, 137

L

letter, 628
list, 645–649
longtable, 172, 173
longtabu, 164

lrbox, 695, 696

M

math, 651
matrix, 287
medaglione, 418, 419
minibox, 560
minipage, 146, 430, 434, 560, 633, 683, 695–697
multicols, 339, 421, 423
multline, 275, 276, 280

P

picture, 177–179, 181, 183, 185, 186, 189, 191, 193, 322, 331, 400, 443, 641, 701
pmatrix, 287

Q

quotation, 134
quote, 133, 134

R

riquadro, 418, 695, 696

S

sidewaystable, 167
sintassi, 418, 419
sloppypar, 568
split, 273–275, 280
subequations, 274, 279, 280

T

tabbing, 629, 681
table, 144, 154, 160, 161, 175–177, 193, 471, 677
tabu, 164, 169, 174, 289
tabular, 146, 155, 156, 164, 177, 191, 628, 629, 682, 683
*tabular**, 156, 164, 682, 683
tabularx, 164
thebibiography, 237
thebibliography, 139, 140, 231, 237, 686
theglossary, 245
theindex, 243, 245, 420, 423, 688
tikzpicture, 186, 187, 189, 701
titlepage, 469, 644, 660

V

verbatim, 135, 650
verse, 134
Vmatrix, 287
vmatrix, 287

W

wrapfigure, 471
wraptable, 471

X

xy, 291

Indice delle classi

A

arstexnica, 83
article, 93, 112, 465, 635, 638

B

beamer, 113, 323, 324
book, 19, 36, 93, 108, 111–114, 118–120, 127, 137, 140, 168, 246, 298, 300, 421, 446, 460, 466, 468–470, 479, 480, 636, 638, 647–649, 678, 679

L

layaureo, 298
letter, 112, 628, 635, 638
ltnews, 112
ltxdoc, 112, 638
ltxguide, 112

M

memoir, 115, 116, 154, 246, 248, 299, 388, 439, 449, 460, 461, 464, 466, 468, 469, 477
minimal, 112, 565, 638

N

ncc, 116

O

octavo, 36, 113, 298, 451, 454
ottavo, 298

P

proc, 112, 638

R

report, 93, 112, 118, 638

S

scrartcl, 114
scrbook, 114
scrlttr2, 114, 115
scrreprt, 114
slides, 113, 321, 322, 638
suftesi, 36

T

toptesi, 118

Indice dei file

Estensioni

,map, 381
,tfm, 377
.afm, 373, 383
.aux, 103, 685, 687
.bat, 88, 495
.bbl, 234, 685
.bib, 240
.bmp, 218
.bst, 232, 239
.dat, 572
.def, 587
.doc, 500–503, 506–510, 512, 598
.docx, 512
.dtx, 415, 435
.dvi, 245, 394, 395, 639, 685
.engine, 215
.eps, 210, 216, 218, 219, 222, 226
.fd, 378, 387, 388
.fmt, 79
.fodt, 516
.gif, 612
.glo, 246, 685
.gls, 246
.gst, 248
.icc, 489
.icm, 489
.idx, 242, 246, 247, 685
.ind, 243, 246, 247, 685
.ini, 81
.ist, 247
.jpg, 218, 219, 224
.lof, 635, 685
.log, 76, 80, 90, 103, 228, 259, 390, 558, 559, 614, 617, 620–622, 680, 685, 688
.lot, 635, 685
.ltx, 80, 102, 685
.lyx, 510
.map, 378
.mf, 89, 396
.mp, 212
.mps, 90, 212
.odt, 512
.otf, 373
.pdf, 210, 212, 216, 218, 219, 222, 226, 245, 395, 501, 579, 598, 639
.pfa, 383
.pfb, 373, 381, 383, 384, 491
.png, 218, 219, 612
.prj, 68
.ps, 210, 218, 226, 395, 496, 639
.rtf, 501, 507, 512
.scriv, 517
.sty, 126, 331, 378, 405
.svg, 210, 217, 222
.sxw, 512
.tex, 46, 47, 56, 77, 80, 91, 102, 223, 494, 496, 507–510, 512, 578, 579, 581, 585, 587, 590–593, 598, 599, 612, 628, 682, 685, 706
.tfm, 373, 382, 383, 396
.toc, 635, 686

.ttf, 373, 384
.txt, 500, 501
.vf, 377, 383
.vpl, 383
.xml, 512
.xmpdata, 494
.zip, 76

A

auctex, 45
Autorun.inf, 42

B

beamerexample2.article.pdf, 324
beamerexample2.beamer.pdf, 324
beamerexample5.pdf, 324
beameruserguide.pdf, 324
Bibliografia.bib, 239
bk10.clo, 446, 456, 648
bohahyph.tex, 571
book.cls, 127, 415

C

classic.ist, 247
clsguide, 113
cmr10.300pk, 382

D

dvipdfm.map, 374, 384
dvipsnam.def, 703

E

ECI-RGB.V1.0.icc, 489, 494
encguide.pdf, 585
endnote.tex, 145
endnotes.sty, 145
esempio1.dvi, 97
esempio1.tex, 97
euler.pdf, 365

F

fntguide.dvi, 704
fntguide.pdf, 364
fontmath.ltx, 365
fonts/map/, 386
fourier-doc-en.pdf, 365

G

gfsartemisia, 310
gfsbaskerville, 310
gfsbodoni, 310
gfscomplutum, 310
gfsdidot, 310
gfsporson, 310
gfssolomos, 311
gglo.ist, 248
gloss-greek.ldf, 571
grfguide.pdf, 701
GuidaGuIT.tex, 494
GuidaGuIT.xmpdata, 494

H

hyph-it.tex, 564
hyph-quote-it.tex, 564, 565

I

ind.dvi, 242
it_IT.aff, 67
it_IT.dic, 67
italian.ldf, 570, 617
ithyph.tex, 564

L

language-local.dat, 572
language-local.def, 572
latex.fmt, 80, 84, 245, 337
latex.ltx, 79, 81, 399, 649
latin.ldf, 570
latin1.def, 585, 586
libretto.pdf, 442

loadhyph-it.tex, 564
local.map, 386, 397

M

MacroGuida.sty, 36, 187, 402, 410, 418, 427
makebst.tex, 233, 237
makeindex.dvi, 242
mathdesign-doc.pdf, 365
Mathmode.pdf, 295
memdesign, 116
memman.gst, 248
merlin.mbs, 233
mf, 382
microtype.cfg, 339
miofont.mf, 382
missfont.log, 90
mt-MinionPro.cfg, 339
mymacros.sty, 127, 421, 424

N

neohellenic, 310
nuovateoria.tex, 104

O

ot1enc.def, 589

P

pag.fd, 318
pagk8r.pfb, 318
PDA_def.ps, 496
pdf2pdfa, 496
pdf2pdfa.bat, 495
pdflatex.fmt, 245
pdftex.map, 318, 374, 377, 384
pdfx.sty, 494
pict2e.cfg, 178
plain.bst, 232
plain.tex, 79, 81, 82, 399
protext-setup-en.pdf, 42
ps2pdfa, 496
ps2pdfa.bat, 496
ps4pdf.engine, 215
psfonts.map, 374, 384
psnfss2e.pdf, 375

S

scrguien.pdf, 464
segnature.pdf, 442
srcguide.gst, 248

T

t1enc.def, 586, 588
t1lmr.fd, 388
tds.pdf, 48, 52
tex, 517, 518, 520, 522, 524
texlive2013-20130530.iso, 44
TeXworks.app, 76
tiger.eps, 216
tttomia.cls, 127
txfonts, 311
txfontsx, 311
txt, 515–517

U

uni3.map, 381
uni3.pfb, 381
uni3.tfm, 381
unsrt.bst, 232
updmap.cfg, 385, 386, 396, 490
updmap.map, 397

Indice dei pacchetti

A

afterpage, 167, 176, 427, 428
alltt, 640, 650
amscd, 290
amsfont, 641
amsfonts, 254, 271, 272, 364, 640
amsmath, 117, 173, 174, 251, 254, 257, 259, 261, 263–265, 269, 271, 273, 277, 280–282, 284, 286–288, 290, 295, 363, 365, 370, 483, 484, 631, 640, 652–654, 704
amssymb, 271, 640, 641
amsymb, 363
array, 157, 159, 168–170, 173, 431, 683
auctex, 65, 70, 75
auto-pst-pdf, 185
avant, 318

B

babel, 26, 27, 94, 99, 102, 116, 122, 132, 133, 149, 265–267, 290, 291, 299–301, 307, 315, 328, 334, 367, 369, 400, 404, 410, 415, 417, 422, 505, 558, 562, 569, 571, 617, 635, 640, 651–653
beamer, 113, 323–325, 329, 331, 332
bgreek, 300
biblatex, 232, 239, 240
bm, 286, 363
bohairic, 371
booklet, 442, 443
booktabs, 161, 164

C

calc, 658
caption, 114, 154, 155, 464
circuitikz, 186
ClassicThesis, 36, 115, 117, 479
color, 322, 464, 475, 490, 640, 702
combelow, 589
cool, 483
coptic, 313, 371
crop, 443
csquotes, 239
curve2e, 328, 331

D

dcolumn, 168

E

ecclesiastic, 371
edmac, 312
edstanza, 312
eledmac, 312
endnotes, 145, 312
epsconversion, 216

epstopdf, 216
etex, 694
etoolbox, 318, 423, 560, 617, 667, 670–672
euler, 365
exercise, 149

F

fancyhdr, 114, 446, 447, 459, 464, 642
fixltx2e, 468, 675
float, 176, 177, 470, 472
floatrow, 177
fontenc, 328, 557, 579, 581, 706
fontspec, 86, 318
fourier, 365, 376, 410, 459
frontespizio, 117, 118

G

geometry, 114, 460, 461, 464, 643
ghostscript, 54
glossaries, 244, 245
glossary, 245
graphics, 475, 641, 702
graphicx, 166, 209, 212, 221, 330, 403, 431, 475, 490, 641, 702
guit, 94, 328

H

hyperref, 240, 406, 493, 494, 510
hyphenator.js, 557

I

IEEEtrantools, 173
ifthen, 471, 641, 667–669, 671
ImageMagik, 220
imakeidx, 248, 249, 641, 688, 689
imakeidxx, 247
indentfirst, 458
indextools, 247–249, 641, 688, 689
inputenc, 59, 60, 75, 328, 505, 557, 579, 581, 591, 593, 598, 634, 682, 706

K

koma-script, 114
kpfonts, 305, 376
kuvio, 290

L

latexsym, 254, 641
layaureo, 119, 120, 452, 453, 480
layout, 122
ledmac, 312
lettrine, 402, 403, 459
longtable, 167, 168
ltxsymb, 271

M

MacTeX.pkg, 46
makeidx, 243, 641, 688
mathdesign, 365, 367, 375
mathenv, 173
mdwtab, 173
metre, 313
mfnfss, 306
microtype, 333, 334, 338–341, 565, 568, 569, 602
morefloats, 676, 680
multicol, 421, 643, 675
multirow, 157

N

natbib, 232, 233, 240
newpx, 375
newpxmath, 360, 361, 365
newpxtext, 360, 365
newtx, 375
newtxmath, 360, 361, 365, 375
newtxtext, 360, 365, 375

O

opcit, 312

P

pdfcomment, 214
pdfpages, 215, 442, 443
pdfscreen, 323
pdfslide, 323
pdfx, 491, 493, 494
pgf, 185, 186, 188, 209, 323, 324, 328, 701
pgfplots, 91
pict2e, 122, 178, 182, 183, 185, 326, 328, 331, 641, 701
Pifonts, 375
poemscol, 312
polyglossia, 27, 315, 571
powerdot, 323, 332
ppower4, 323
prosper, 323, 332
psfrag, 189
pst-pdf, 185
PSTricks, 185, 214, 217, 332, 701
PSTriks, 215
pxfonts, 359, 361, 365, 375

R

rotating, 167
rotfloat, 177

S

sapthesis, 117
sectsty, 464
seminar, 323
showidx, 641
SIstyle, 267
SIunits, 267
siunitx, 164, 267, 268
suftesi, 117, 299
supertabular, 167, 168
svg, 217, 222

T

tabmac, 312
tabu, 164, 168, 174, 288, 289
tabularx, 162, 164, 168
technica, 299–301
TesiClassica, 117
TesiModerna, 117
testhyphens, 558
teubner, 301, 302, 308, 309, 311–313, 370
texpower, 323
textcomp, 367, 510, 580, 581, 586, 587
TikZ, 91
tikz, 187, 290
titlesec, 464, 465
tocloft, 407, 408, 410, 464
topcapt, 168
TOPtesi, 117, 118
trace, 619, 620, 624, 625
txfonts, 359, 361, 365, 375
type1ec, 638
typearea, 114, 115, 454, 460, 464

U

unicode-math, 86
units, 267
utopia, 365

V

varioref, 122, 141

W

widetable, 164
wrapfig, 471, 472

X

xcolor, 164, 323, 324, 490, 702
xypic, 133, 290, 291

Indice dei programmi e delle distribuzioni

Simboli

TEXShop, 73

A

AbiWord, 507, 508, 510, 512
Acrobat Professional, 488, 493, 498
Adobe Acrobat, 55, 221
Adobe Reader, 55, 69, 221, 490, 501
afm2tfm, 383
Aleph, 86, 87
aleph, 91
Anteprima, 55, 221, 511
Aquamacs, 75, 76, 585, 591, 592, 598
Aquamcas, 592
arara, 90
arlatex, 91
asy, 91
Asymptote, 91
auctex, 71, 76
autoinst, 383, 384

B

bash, 397
BibDesk, 46, 89, 233
biber, 89, 141, 231, 232, 240, 244, 687
BIBTEX, 72, 89, 141, 231, 232, 234–240, 244, 312, 406, 685–687
bibtex8, 89
bibtexu, 89

C

Character map, 57
charco, 597
context, 91, 513, 601
context-mk-iv, 590
convert, 220
CygWin, 69, 382, 507, 597

D

dbcontext, 513
dblatex, 512, 513
Draw, 511
dvipdfm, 78, 89, 214, 215, 332, 384, 487, 490
dvipdfmx, 89, 214, 215, 374, 381
dvips, 54, 55, 78, 89, 178, 185, 189, 214, 215, 332, 374, 381, 384, 395, 490, 513, 703

E

ebb, 214
elatex, 82

emacs, 45, 65, 68, 70, 71, 75, 76, 512, 585, 592
eps2jpg, 218
eps2pdf, 218
eps2png, 218
epspdf, 218
epstopdf, 89, 218
ε-TEX, 409
etex, 80, 82, 86, 91, 245, 361, 431, 455, 658, 663
EurKey, 58
Excel, 200
excel, 511

F

fmtutil, 83, 90
fmtutl-sys, 90
FontForge, 86, 377, 381, 383
fontinst, 383

G

gedit, 72
gedit-LaTeX-plugin, 72
ghostscript, 42, 54, 218, 487, 488, 493, 495, 496
ghostview, 42, 54, 218
gimp, 55, 219, 220
gnome, 72
gnuplot, 195, 198, 200
GraphicConverter, 220
gs, 214, 216
gv, 54
gview, 54, 55, 218, 220, 225, 226, 228

I

iconv, 594, 597
ImageMagik, 220
InDesign, 4
inkscape, 217, 490

J

JabRef, 233
Jabref, 89
java, 397
jpegtops, 55, 219

K

Kile, 45, 69–73, 76
kpathsea, 86
kpseaccess, 91
kpsepath, 91
kpsereadlink, 91
kpsestat, 91
kpsetool, 91
kpsewhere, 91
kpsewhich, 48, 91, 385
kpsexpand, 91
Kword, 507, 508, 510, 512

L

lamed, 91
latex, 46, 65, 78–81, 91, 214–216, 394, 405, 498, 509, 510, 513
LaTeXiT, 46, 47
latexmk, 90
LED, 68, 69, 84
Libre Office, 518
LibreOffice, 577
lua, 86, 87
lualatex, 87, 88, 91, 503
luatex, 85–87, 91, 503, 590, 601
LyX, 77, 498, 509, 510, 512

M

MacTEX, 45, 88, 233
makeindex, 89, 242–249, 401, 685, 688, 689
Mappa dei caratteri, 57
Marked2, 517
METAPOST, 86, 189, 210, 212, 214, 216, 219

mftrace, 382, 383, 396
MiKTEX, 42, 43, 48, 49, 51–53, 66–70, 88, 89, 97, 122, 247, 377–380, 386, 387, 392, 394, 569, 570, 613, 667
MiKTeX Maintainance, 53
MiKTeX Settings, 53
mktexlsr, 91
mllatex, 91
mltex, 91
mptopdf, 89, 219
MultimarkdownComposer, 517

N

NeoOffice, 557
Notepad, 69, 396

O

Omega, 86
Open Office, 518
Open XML converter, 512
OpenOffice, 557
OpenOffice.org, 511

P

Pages, 518
Paint, 219
Pandoc, 499, 515
patgen, 572, 573
pdfcrop, 90, 221
pdfelatex, 82
pdfetex, 80, 82
pdfimport, 511
pdfLaTeX, 329
pdflatex, 46, 66, 71, 75, 78–81, 86–88, 91, 187, 215–218, 222, 305, 309, 311, 313, 318, 319, 334, 353, 371, 394, 395, 405, 442, 488, 489, 493, 495, 498, 500, 505, 506, 508–510, 513, 558, 560, 568, 572, 578, 579, 586, 587, 590, 598, 600, 612, 641, 685, 688
pdftex, 79, 80, 82, 84–87, 91, 128, 180, 189, 215, 216, 245–247, 334, 337, 361, 374, 384, 399, 431, 443, 455, 456, 471, 487, 490, 569, 601–603, 605–607, 658, 663–666, 692
pdftex.exe, 247
pdftops, 219
perl, 397
potrace, 382
Preview, 55, 221, 511
ProTEXt, 42, 68
ps2pdf, 55, 78, 215, 218, 332, 495
ps4pdf, 89, 215, 217
pstopdf, 89
ptex, 91
Python, 72
python, 382, 397

Q

QuarkXPress, 4

R

rtf2latex2e, 507

S

Saxon, 512
Scribus, 3, 4
Scrivener, 517–520, 522, 524
Skim, 55, 75, 76, 511
StarOffice, 511
SumatraPDF, 42, 66

T

Task Manager, 614
tex, 79, 80, 82, 84, 86, 91, 215, 246, 399, 455, 456, 570, 601

TeX Live Manager, 53
TeX Live Utility, 46
TeX4ht, 499
texdoc, 48, 91, 476
TEX Live, 41–46, 48–54, 66, 68, 70, 71, 216–218, 244, 247, 248, 295, 299, 306, 313, 360, 368, 377–380, 384–387, 392, 397, 477, 569, 571, 572
TeXmacs, 77
TeXmaker, 45, 66, 600
Texmaker, 66, 68, 70, 72, 73, 77, 585
TeXnicCenter, 68–70
Texpad, 524
TeXShop, 66, 67, 72, 74, 109, 524
TEXShop, 46, 64, 72, 73, 75–77, 84, 97, 215, 219, 314, 315, 585, 588, 591, 592, 598, 599, 613, 614, 616
TeXStudio, 45
TeXstudio, 42, 65, 66, 68–70, 72, 77, 109, 233, 585, 592, 600
TextEdit, 512, 518
TextPad, 69
TeXtures, 77
TextWrangler, 76
TeXWorks, 524
TeXworks, 45, 46, 66, 67, 72, 75–77, 315, 585, 588, 591–594, 597–600, 613
tlmgr, 44, 45, 53
ttf2tfm, 384

U

uninstl.bat, 43
updmap, 90, 379, 384, 490
updmap-sys, 90, 384, 385

V

vim, 70, 396
vptovf, 383

W

Win-iconv, 597
WinCDEmu, 43
WinEdt, 68, 69, 219, 503
Word, 518
Word2LyXMacro, 509
word2tex, 507
Writer, 509, 511, 518, 577
Writer2LaTeX, 509, 511
Writer4LaTeX, 509

X

xdvi, 73, 97, 374, 394
xdvipdfmx, 89
xelatex, 27, 86–88, 91, 309, 310, 314–316, 318, 319, 503, 558
Xemacs, 71
xetex, 86, 91, 503, 590, 601
xfig, 198
xindy, 89
xpdf, 86

Y

YAP, 97, 374, 394